A BEGINNER'S GUIDE TO SIMPLE PROGRAMMING

FIRST STEPS WITH YOUR BBC MICRO

CAROLYN HUGHES

FONTANA

Rob

First Steps with Your BBC Micro

About the author

Carolyn Hughes first entered the world of computers when her family acquired a Sinclair ZX80. They have since added to their collection a ZX81, a ZX Spectrum and a BBC Micro.

She has three children and runs a computer club at the Junior School her youngest son attends. Although she is particularly interested in children's involvement with computers, both in education and just for fun, she realises that adults, too, often need a non-technical guide to take them through their first encounters with computers.

As well as being the author of Armada's *First Steps with Your Spectrum*, she also writes educational software.

CAROLYN HUGHES

First Steps with Your BBC Micro

with illustrations by
BRYAN READING

A Fontana Original

First Steps with Your BBC Micro was first published
in the U.K. in 1984 by Fontana Paperbacks,
8 Grafton Street, London W1X 3LA.

Text © Carolyn Hughes 1984
Illustrations © Bryan Reading 1984

Printed in Great Britain at
the University Press, Oxford.

CONDITIONS OF SALE
This book is sold subject to the condition
that it shall not, by way of trade or otherwise,
be lent, re-sold, hired out or otherwise circulated
without the publisher's prior consent in any form of
binding or cover other than that in which it is
published and without a similar condition
including this condition being imposed
on the subsequent purchaser.

Contents

1 What your micro can do. 9
(with your help).

2 Getting started. 12
Tuning the TV.
The prompt and the cursor.

3 Talk to your micro. 15
The keyboard. PRINT.
Quotation marks. RETURN.

4 Mistakes. 18
The DELETE *key. Error messages.*

5 Now for some numbers. 20
Add, subtract, multiply and divide.
Priorities and brackets.

6 A program. 24
Numbering your instructions. RUN.
Blank lines, and spaces. LIST.

7 Another program. 29
NEW, BREAK *and* OLD.

8 Lower case letters. 31

9 Modes. 32
Different styles of printing on your screen.
Moving the display up or down. CLS.

10 A two-way conversation. 36
INPUT *your answer.*

11 Boxes for numbers. 38
Number variables.

12 More about printing. 41
Semi-colons to neaten up your numbers.

13 Strings and characters. 45

14 Boxes for strings. 46
String variables.

15 Another way into the boxes. 49
When to use LET.

16 Names for your variable boxes. 53
What you can call them, and what you can't.

17 Making changes. 56
Using the editing keys.

18 A decision for your micro. 60
IF . . . THEN . . . ELSE.

19 Think of a number. 63
Random numbers with RND.

20 The computer does it again. 66
REPEAT . . . UNTIL.

21 Think of a word. 70
The computer chooses a word with RND.

22 Loops for counting. 74
FOR . . . NEXT *loops for counting and waiting.*

23 Characters and codes. 79
ASCII *codes and* CHR $

24 Your own characters. 82
How to design and program your own aliens and planes with VDU 23.

25 Choose your spot on the screen. 88
Placing it where you want it with TAB.

26 Your characters move. 93
Up, down and across.

27 Colours. 101
COLOUR *in the foreground and background.*

28 Sounds. 106
Music, and other sounds.

29 GET $ *and* INKEY $. 112
The computer reads the keys, (and sometimes waits).

30 Draw and paint. 119
High resolution graphics.
MOVE, DRAW *and* PLOT 85.
GCOL0 *for colour.*

31 Planning a program. 126
Writing it down. REM *to remind you.*

32 Saving your program on cassette. 133
How to SAVE, LOAD, CHAIN *and* *CAT.

33 Games. 140

34 You could do it like this. 155
Suggestions on the 'For you to do' sections.

Index 167

MODE details 170

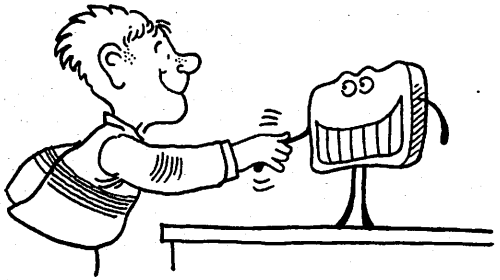
Acknowledgements

I should like to express my thanks to Pam, for her assistance in typing the manuscript, and to Catherine and Alf for carefully checking it through and trying out the programs.

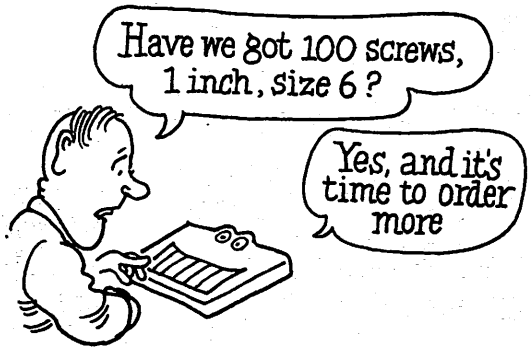
CHAPTER 1

What your Micro can do

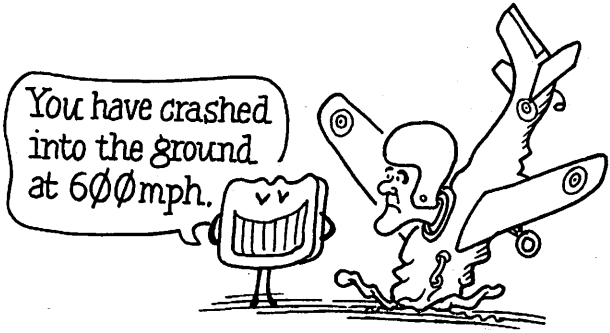
Welcome to the growing band of computer programmers! As computers are playing a greater part in our daily lives, so more people are finding out just what they can do, and how to control them.



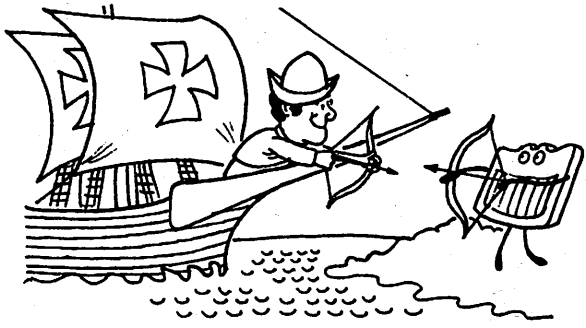
Although your BBC Micro may look small, it has a large memory, and can work things out extremely fast. That is why people are using small computers to do jobs which humans would find very slow or boring.



Computers are being used to record hospital patients' blood pressure, and to help handicapped people to enjoy life more fully. In factories and shops they can keep records of what items are in stock, and let the humans know when it is time to order more. At home, you can find out what it is like to pilot an aeroplane, or try to find your way out of a monster-infested maze.

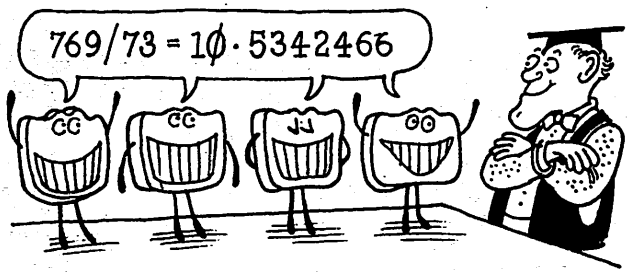


BBC Micros are helping in many ways with education in Primary and Secondary schools. Suppose you are studying the fifteenth-century voyages of discovery, for example. With the help of the computer, and a suitable program, you can experience the storms and shipwrecks, discover new lands, and search for gold. The computer plans your adventures according to the decisions you make, so you can find out what it was like to be an explorer like Magellan or Christopher Columbus.



It may sound as though computers can do a lot of things better than we humans. They can, but only if we first show them how.

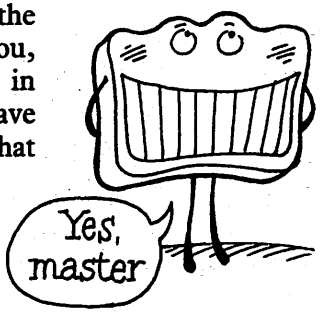
When your BBC Micro was being made, it was taught Maths. It can do addition, subtraction, multiplication, and also other more complicated mathematical calculations.



It was also taught a language, called BASIC. It knows about a hundred words of this language, many of which are the same as English words, such as PRINT, IF, REPEAT. These are known as BASIC 'keywords'.

The computer remembers its Maths and BASIC in a part of its memory called the ROM. There is another part in the memory called the RAM, where it stores the instructions you, the user, give it. One of the differences between the ROM and the RAM is this. When you switch off your BBC Micro, the RAM memory is wiped clean, so it forgets the instructions you gave it. The ROM memory, however, is permanent, so it does not forget how to do Maths or understand BASIC.

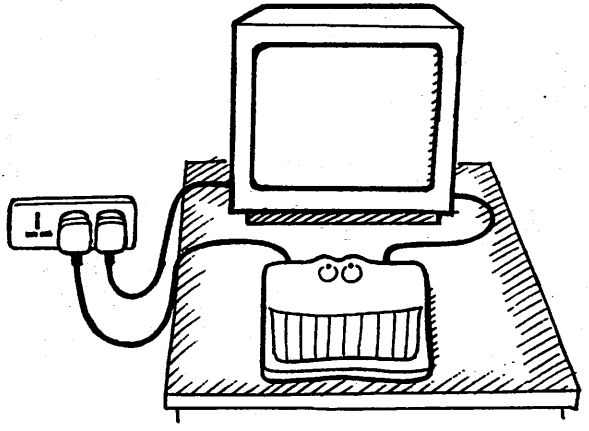
This book aims to introduce the most useful BASIC keywords to you, and show you how to use them in writing programs. You will then have control over your BBC Micro, so that it does what *you* want it to.



CHAPTER 2

Getting started

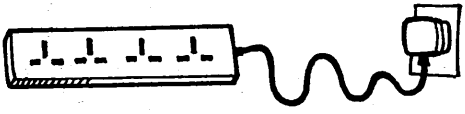
When speaking to your computer, you type your message in at the keyboard. Before you can receive its side of the conversation you must connect your micro to a television set. Most T.V. sets are suitable. If you have a colour set, so much the better. If you only have black and white, don't worry – your programs will still work, but the colours will appear as shades of grey.



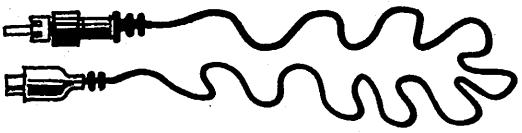
Put your micro on a table in front of the T.V. screen, and near to an electric point. Both the computer and the T.V. need to be plugged in to the electricity, so if there is only one electric point, you wwill need an adaptor so that the two of them can share the electricity supply.

Later on you will want to use a cassette recorder too, for storing your programs, so if you are buying an adaptor it

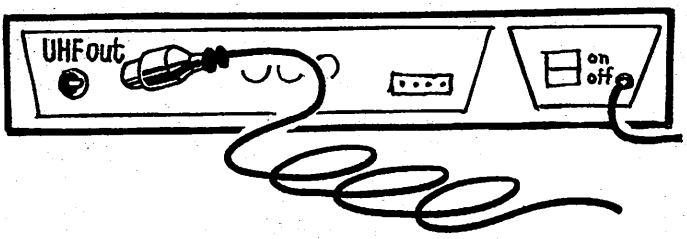
would be worthwhile getting one which can take three or more plugs.



With your BBC Micro comes an aerial lead, which looks like this:

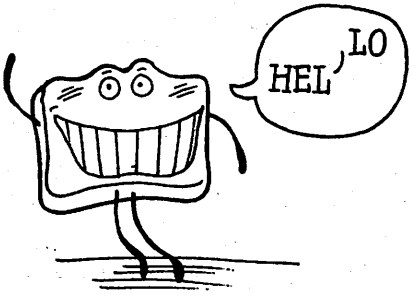


The pointed end fits into the socket on the back of the computer marked UHF out:



The other end goes into the aerial socket of your T.V. If the television has two aerial sockets, use the one marked UHF.

With the computer and T.V. plugged in, and the aerial lead in place, you are ready to start. Switch on the electricity, and put the BBC Micro's switch to on. It will greet you with a two tone sound.



Now switch on the T.V. Turn the volume right down. You will probably have a snow-storm on the screen, so you will need to tune in the T.V. If your set has several tuning knobs, one for each channel, choose one which you do not use for watching T.V. programmes. Turn it until the screen shows a clear message like this:

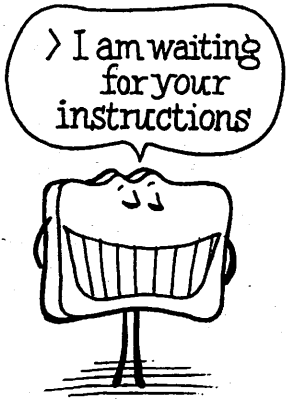
```
BBC Computer 32K

BASIC

>_
```

If you have a model A computer, rather than a model B, it will say 16K instead of 32K. This tells you how large a memory your computer has. As you can see, the model B's memory is larger, and so it can handle longer programs.

You may be wondering what > is, and why that line _ is flashing away on the screen. The > is called the 'prompt'.



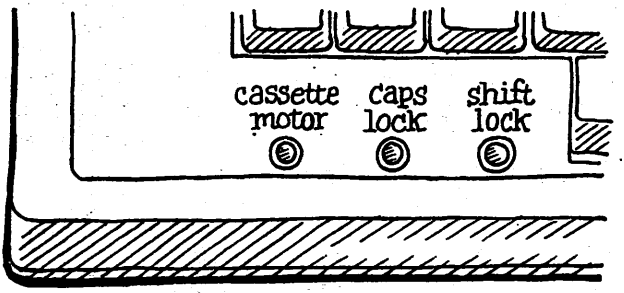
The flashing line is the 'cursor' and shows you where on the screen your typing will appear.

Your micro is now ready and waiting for you to start.

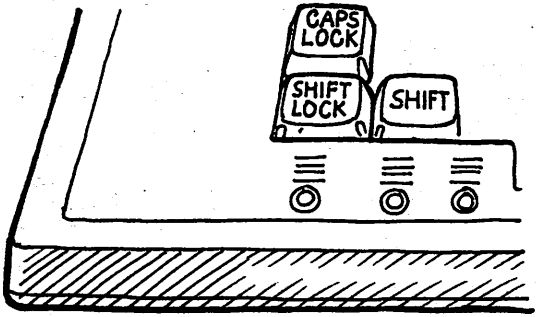
CHAPTER 3

Talk to your Micro

Before you give your BBC Micro its first instruction, have a good look at the keyboard. Near the bottom left-hand corner you will see three red lights, labelled 'cassette motor', 'caps lock', and 'shift lock'.



The cassette motor light only comes on when you have a cassette player attached and working. The shift lock light should be off. If it is on, press the key marked SHIFT LOCK, and it will go out.



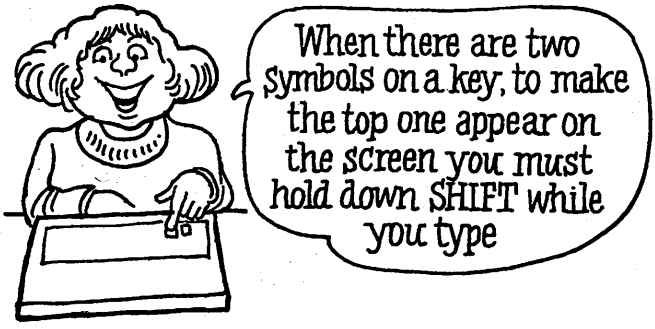
The important light at the moment is the caps lock one. When this is lit, all the letters you type will appear as capitals. The BBC Micro will only understand your instructions if the BASIC keywords are in capital letters. So if your caps shift light is not lit, press the CAPS LOCK key.

Now for your micro's first instruction. We are going to tell it to greet you by printing HELLO on the screen. You will need to type

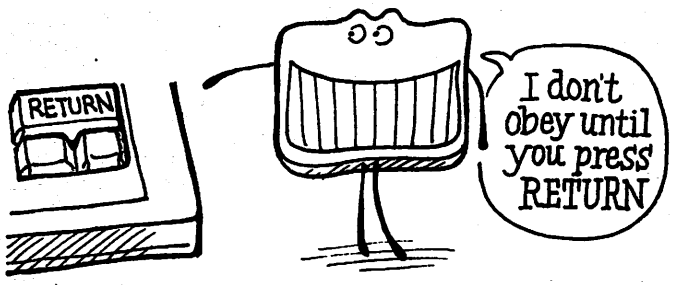
```
PRINT "HELLO"
```

Print is easy. Just find the keys, and press them. Now press the space bar once. It is the long bar below the letter keys.

You will find the quotation marks " on key 2. Before you press the key, put your finger on one of the SHIFT keys and hold it down. Now press the " key. Take your finger off SHIFT and type HELLO. Finish with another set of quotation marks.



Your instruction is now on the screen, but before the micro will obey, you must press the RETURN key.



The computer has obeyed your command and printed HELLO. If it has not done so, turn to the next chapter.

The quotation marks are very important. They tell the computer to print on the screen everything it finds between them.

For you to do

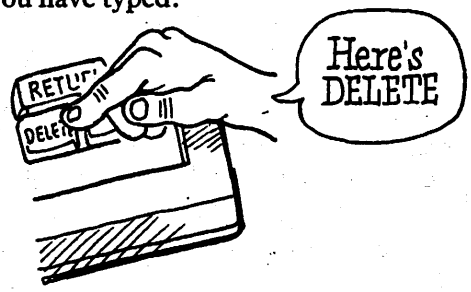
Tell the computer to print your name on the screen. Remember to put quotation marks before and after your name.

You will find suggestions for the 'For you to do' sections on p. 155.

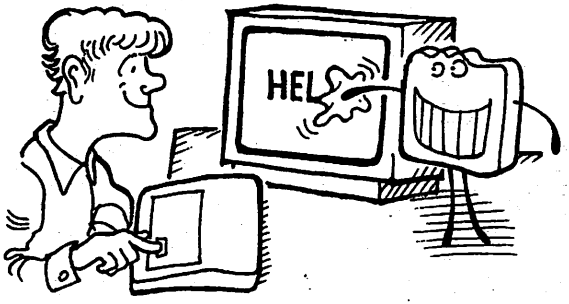
CHAPTER 4

Mistakes

It is easy to press the wrong key when you are typing, even if you are an experienced typist. If you make a mistake in entering an instruction, and notice your mistake before you have pressed RETURN, you can use the DELETE key to rub out what you have typed.



Type in a few letters (any will do). Press the DELETE key lightly. The last letter has disappeared, and the flashing cursor has moved back to take its place. If you hold your finger down on DELETE, the computer will keep rubbing out until the line of letters is gone.



What if you don't notice your mistake and press RETURN?

If you have mistyped a keyword, the computer will not understand, and will come back with the comment 'Mistake'.

The computer has various messages which it will give you, according to the kind of mistake you have made. Some of these you will find helpful. Others you will not understand until you know more about programming.

If you make a mistake and press RETURN, the only remedy is to type in the instruction again.

CHAPTER 5

Now for some numbers

You have used the BASIC keyword PRINT to put words on the screen. Now we will use it to print the answers to some mathematical questions.

Type in

```
PRINT 443 + 689
```

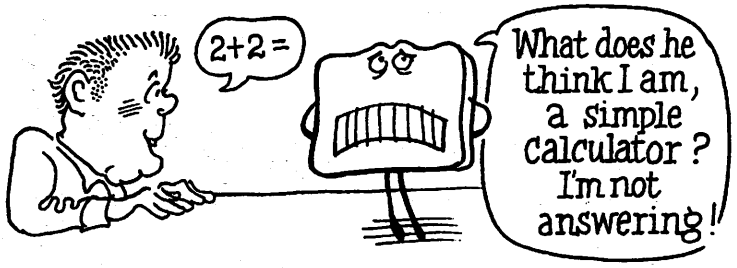
and press RETURN

Quick as a flash the computer comes up with the answer. Notice we did not use quotation marks this time. Do you know what would have happened if you had typed

```
PRINT "443 + 689"        [RETURN]?
```

Try it and see.

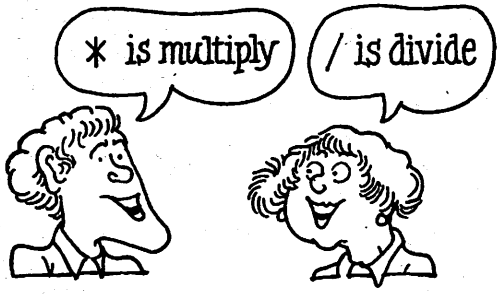
Remember that your computer does not work in the same way as a calculator, and will be offended if you treat it like one!



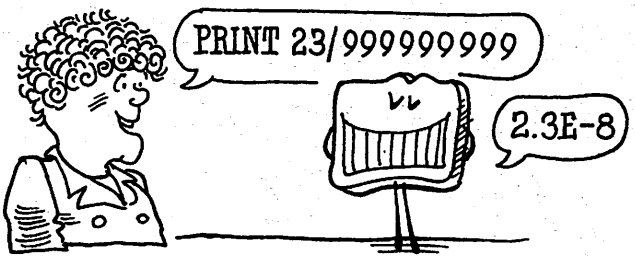
Now give it a subtraction question. You will find the minus sign near the top of the keyboard, on the same key as the equals sign.

```
PRINT 5101 - 3978        [RETURN]
```

The signs for multiplication and division are different from the usual ones.



Give your BBC Micro some * and / questions. It can manage very large numbers, but if the numbers are too large it will give the answer in a way you may not understand.



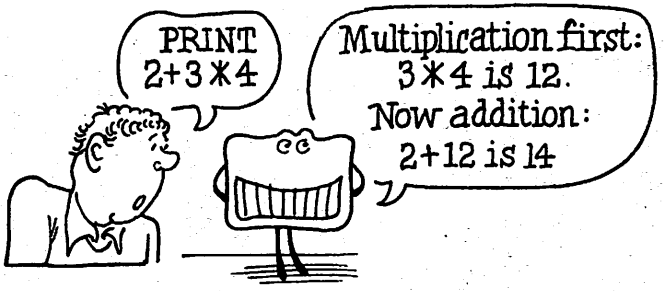
If you were asked to work out 2 + 3 * 4, you would probably start at the left, with 2, add 3, making 5, and then multiply by 4, making 20. Not so the computer!

Type in

```
PRINT 2 + 3 * 4        [RETURN]
```

and see what happens.

Computers have a special order of working out mathematical questions. They do all the multiplication and division first, then the addition and subtraction. So what your BBC Micro did was this:

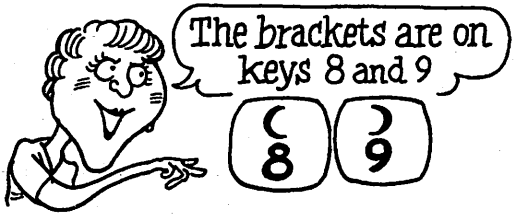


We say that multiplication and division have a 'higher priority' than addition and subtraction.

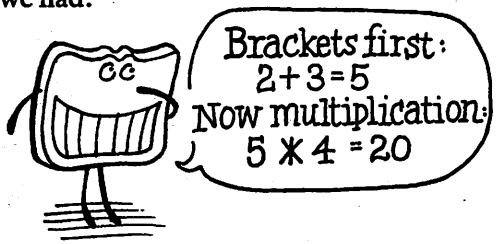
If you want to change the order of working out, put brackets round the part to be done first.

Type in

```
PRINT (2 + 3) * 4        [RETURN]
```



This time we had:

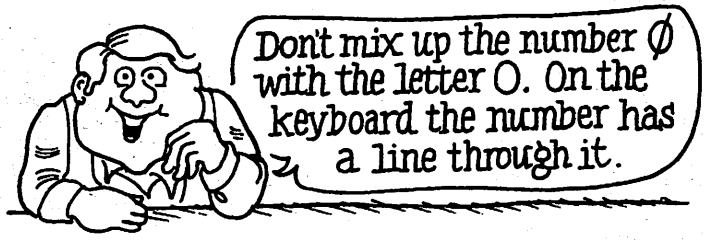


For you to do

Try some questions of your own with + − * and / and see if the BBC Micro gives you the answers you expect. Here is one to start with:

```
PRINT 8 - 2 * 3          [RETURN]
```

By putting in brackets, see if you can make the answer 18.

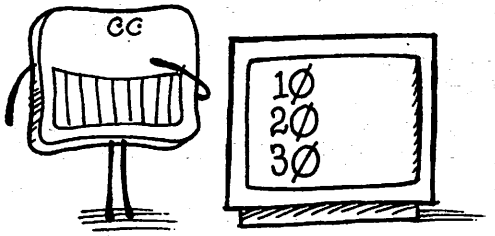


CHAPTER 6

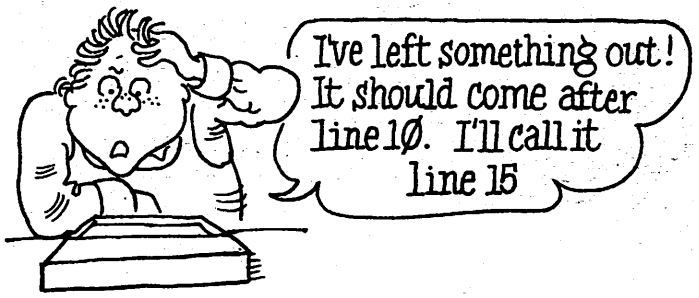
A program

Now the micro is obeying your instructions, but only one at a time. If you put your instructions into a 'program', the computer can carry out a long list of commands. It will also remember them, so that it can repeat the instructions without you having to type them in again.

In a program, we give each instruction a number, so that the computer knows which order to do them in.



We usually number the instructions, or 'lines', in steps of ten, instead of steps of one, so that if we find we have left something out, we can fit it in.



Type in

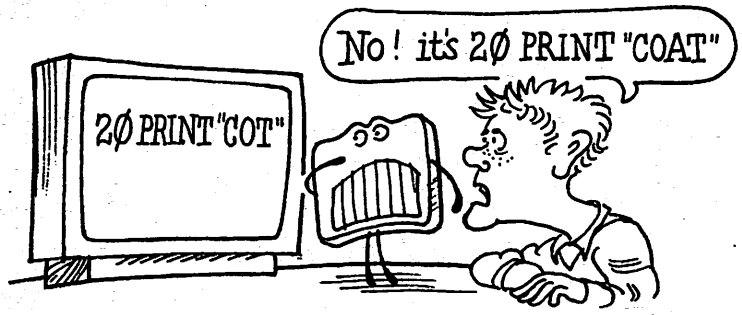
```
10 PRINT "HAT"        [RETURN]
```

The computer does not obey the command straight away. It stores it in its memory for later on.

Now type in

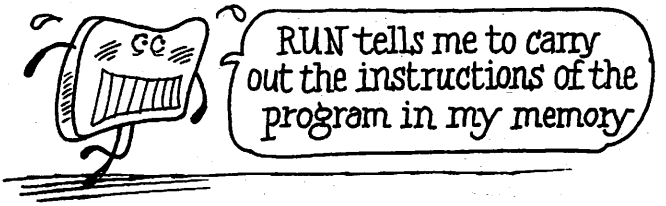
```
20 PRINT "COAT"       [RETURN]
30 PRINT "SHOES"      [RETURN]
```

If you discover after you have pressed RETURN, that the line of the program is not quite as you want it, just type the line again and press RETURN. The computer will put the new line into its memory, in place of the old one with the same number.



If you want to get rid of a program line, rather than replace it with another, just type the line number and press RETURN.

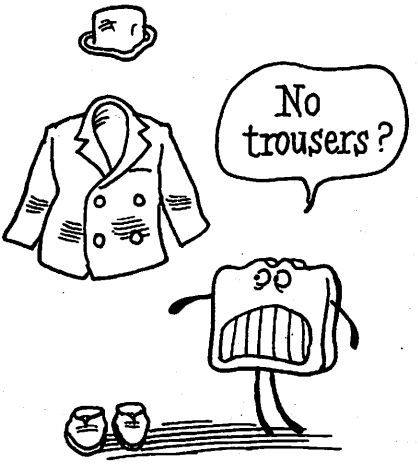
When you are ready for the computer to carry out your three instructions, type RUN, and press RETURN.



After it has obeyed your commands, and printed the three words, the computer still has the program in its memory. Tell it to RUN the program again.

Again the micro prints

```
HAT
COAT
SHOES
```



Now type in

```
25 PRINT "TROUSERS"        [RETURN]
```

The program now has four lines. It would be useful to see them all together on the screen. Type LIST and press RETURN and the computer will give you a 'listing' of your program with the lines in the correct order.

RUN this longer program and you will see that TROUSERS has been fitted in.

A note about spaces

You could have typed your program like this:

```
10PRINT"HAT"
20PRINT"COAT"
25PRINT"TROUSERS"
30PRINT"SHOES"
```

without any spaces. The computer would still understand. The spaces just make it easier for humans to read.

However, spaces inside the quotation marks are a different matter. If the computer sees a space within the quotation marks it leaves a space when printing.

Type in

```
20 PRINT " COAT"        [RETURN]
```

pressing the space bar before the letter C. RUN the program and see what difference this makes.

For you to do
Type in three more lines:

```
15 PRINT
22 PRINT
27 PRINT
```

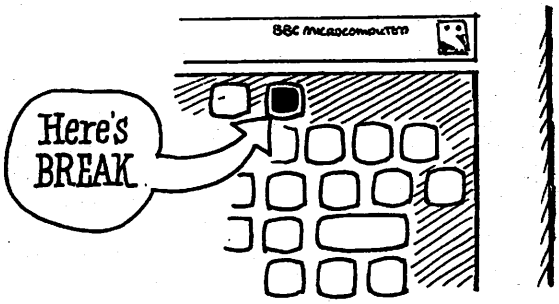
RUN the program and see what the command PRINT, by itself, tells the computer.

CHAPTER 7

Another program

When you have finished with a program, and want to type in a new one, you tell the computer so by typing NEW, and pressing RETURN. You can still see the listing of the program on the screen, but it has gone from the micro's memory. You can check that it has gone by typing LIST and pressing RETURN.

Another way of wiping the memory clean is by pressing the BREAK key (no need for RETURN this time).



As soon as you lift your finger the screen is clear, except for

```
BBC Computer

BASIC

>_
```

What can you do if you press BREAK, and lose your program by mistake? Can you get your program back? Luckily, you can, as long as you have not started to type in another program.

Type OLD and press RETURN, and your old program will go back into the memory. If you wish, you can then tell the computer to LIST it.

For you to do

When you really have finished with the last program, clear your micro's memory with NEW, or BREAK. Now write a program telling the computer to print your name and address on the screen. Remember to put everything you want printed, including commas and full stops, inside the quotation marks.

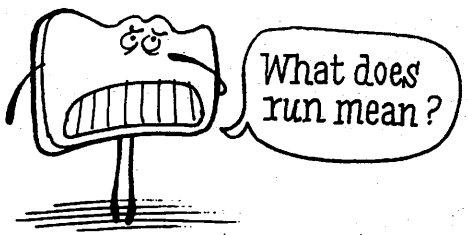
CHAPTER 8

Lower case letters

Perhaps you would like the computer to print your name using small letters, instead of all in capitals. Here is how to do it.

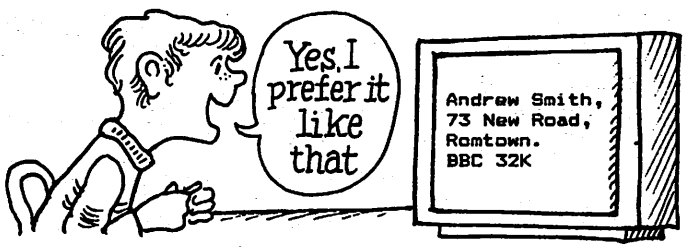
Press the CAPS LOCK key. The red light will go out. All three lights are off and your micro will now work like a typewriter. When you press a letter key, on the screen you will see the small, or 'lower case' version, instead of the capital or 'upper case' one. When you need a capital letter, hold your finger on SHIFT while you type.

Remember that BASIC keywords, like PRINT, RUN, LIST, NEW, OLD, must be in capitals, or the computer will not understand.



For you to do

Try typing in your name and address program again, using a mixture of lower case letters and capitals.



CHAPTER 9

Modes

The BBC computer has a number of different 'modes' or ways of showing your programs on the screen. The Model B has eight ways (called MODE 0 to MODE 7), whereas the Model A has four (MODEs 4 to 7).

Before you can experiment with the modes, you will need a program in the computer's memory. The name and address program from Chapter 8 would be ideal.

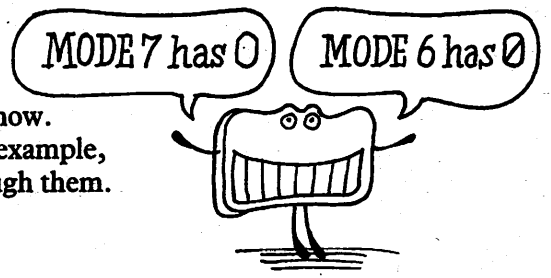
When you first switch on your computer, or when you press BREAK, it is set to MODE 7. This is also known as the Teletext mode. If you have a television which can receive the T.V. companies' teletext transmissions (CEEFAX and ORACLE), you may have noticed that your BBC Micro uses the same style of printing as these do.

Make sure your caps lock light is on, and that you have a program in the memory. Then type:

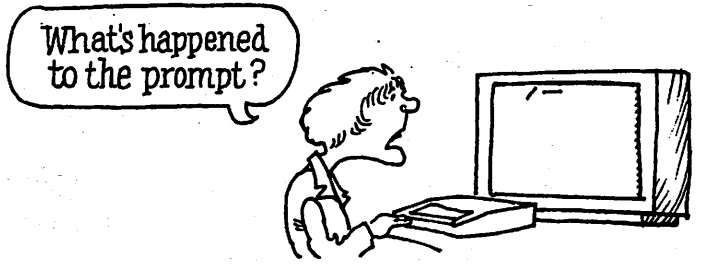
```
MODE 6          [RETURN]
```

The screen will go blank.

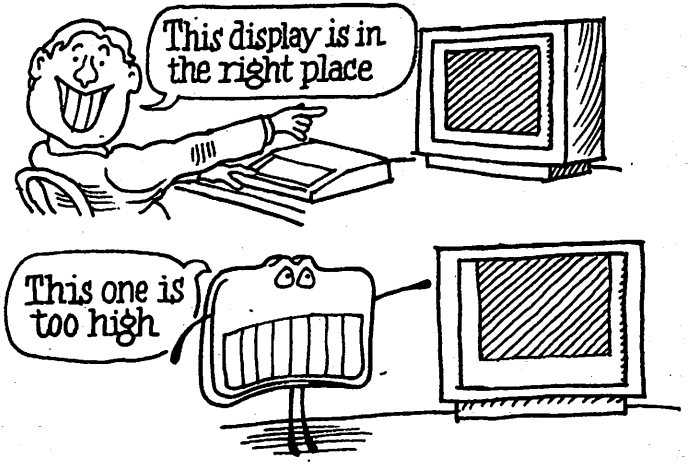
Tell the computer to LIST and then RUN the program. The style of the letters and numbers is slightly different now. The noughts, for example, have slashes through them.



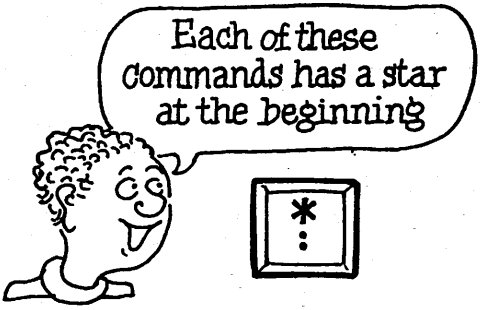
When you change mode, and the screen clears, you may find that the prompt and the cursor have almost disappeared off the top of the screen.



You can put this right by using one of the BBC Micro's 'display moving' commands. The display is the part of the television screen where your typing is shown.

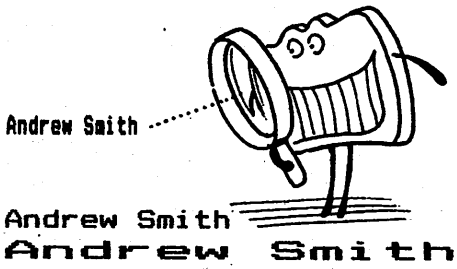


`*TV 255`	moves the whole display down one line.
`*TV 254`	moves it down two lines.
`*TV 1`	moves the display up one line.
`*TV 2`	moves it up two lines.



Type in the command you need and press RETURN. Before the display will actually move, you have to either change MODE (e.g. type MODE 7), or press BREAK. If you press BREAK, you will then have to type OLD to get your program back.

Change to MODE 5 and LIST and RUN your program again. Now try MODE 4, and MODEs 0,1,2 and 3 as well, if you have a Model B computer. You will find that some modes give you very wide letters and others very narrow. Some give more space between each line of writing than others.



You may be saying to yourself, "What's the difference between MODE 2 and MODE 5? They look the same to me". The style of printing is, indeed, the same. The difference is that when you come to use colours, you will find that in MODE 2 you can have sixteen different colours on the screen at a time, whereas MODE 5 allows you only 4. You will find more about this in Chapter 27.

Most people prefer to use MODE 7 when they are typing in

their programs, as the letters are clear and easy to read. Often, however, you will want the computer to change to another mode for the actual 'run' of the program. You can arrange this by instructing the computer, at the beginning of the program, which mode it is to use. Put an extra line into the name and address program, so it reads:

```
 5 MODE 5
10 PRINT "Andrew Smith,"
20 PRINT "73 New Road,"
30 PRINT "Romtown."
40 PRINT "BBC 32K"
```

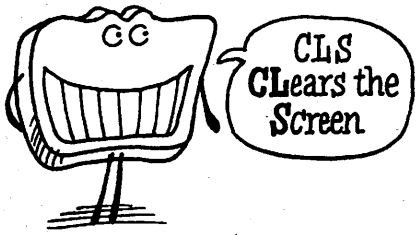
Whenever the computer is told to RUN this program, it will change to MODE 5, no matter which mode it was in before.

After the program has finished running, the computer will still be in MODE 5, until you tell it to change.

Changing mode clears the screen. If you just want to clear the screen without changing mode, you can use CLS.

Type in

```
5 CLS
```



and RUN your name and address program again.

For you to do

Find out, by typing some letters and counting them, how many letters there are on one line in MODE 3, MODE 5 and MODE 7.

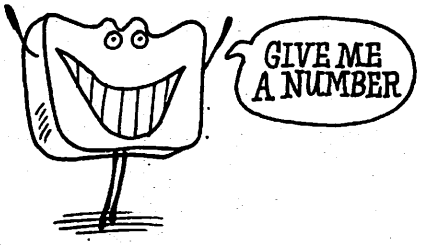
CHAPTER 10

A two-way conversation

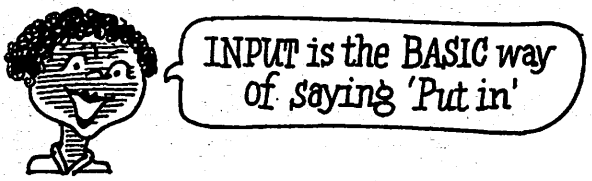
Clear your computer's memory (with NEW or BREAK) and type in this program:

```
10 PRINT "GIVE ME A NUMBER"
20 INPUT NUMBER
30 PRINT "YOU GAVE ME" NUMBER
```

RUN the program, and you will find that, just as you would expect, the computer asks you for a number.



On the next line is a question mark, and the flashing cursor. The instruction INPUT caused this. INPUT tells the computer to print a question mark and then wait while you type something in.



Type in a number and press RETURN. Now the computer moves on to line 30 and prints YOU GAVE ME, followed by your number.

For you to do

Put in two extra program lines — one to tell the computer to clear the screen before it asks for a number, and the second to tell it to leave one blank line before printing YOU GAVE ME. . .

You could number them line 5 and line 25.

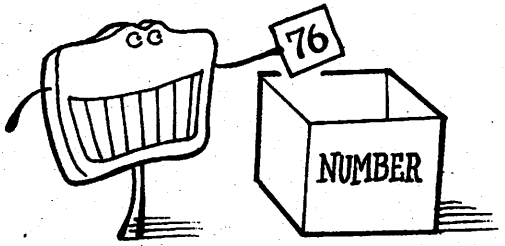
CHAPTER 11

Boxes for numbers

You may have guessed that when you typed in a number, in the 'give me a number program', and pressed RETURN, the computer stored that number away in its memory.

This is how it happened. In its memory, your BBC Micro has a part specially reservd for storing numbers. This section of the memory is like a collection of boxes, waiting to be filled.

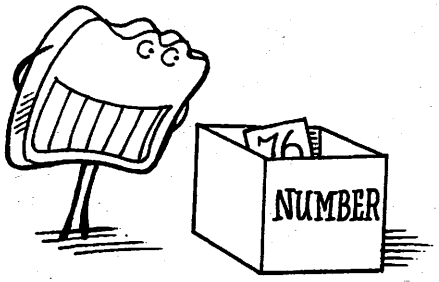
When it saw INPUT NUMBER, the computer labelled one of the boxes with the word NUMBER, and waited for you to type. It then put into the box the number you typed.



In line 30, when it was told to print NUMBER, it looked into the box with that name and printed the number it found there. Even though the program has now finished, your number is still in the box. Type in

```
PRINT NUMBER          [RETURN]
```

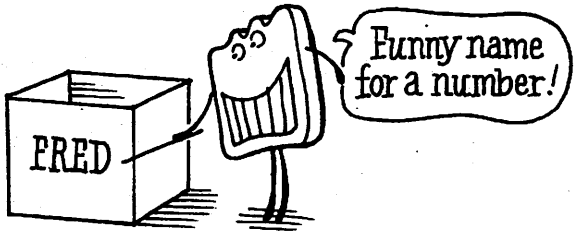
to check this.



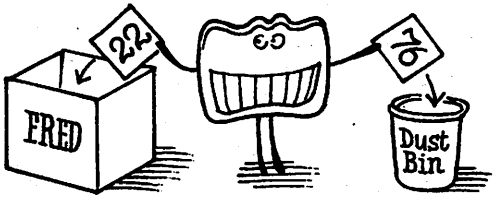
You don't have to call the box NUMBER. You could call it FRED if you wanted. Type in lines 20 and 30 again, this time putting FRED, instead of NUMBER.

```
20 INPUT FRED
30 PRINT "YOU GAVE ME" FRED
```

RUN the program and you will see it works just as well.



The boxes for storing numbers are called 'number variables' because you can change, or vary, the numbers they hold.



For you to do

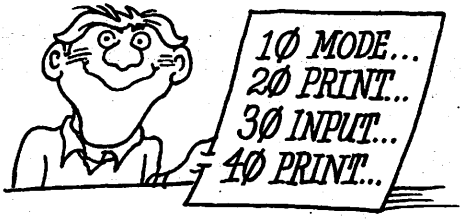
Write a program in which:

The computer goes into MODE 7

It asks you for the number of your house.

You type it in, and the computer stores it in a box called HOUSE.

It replies with YOU LIVE AT HOUSE NUMBER and then your number.



CHAPTER 12

More about printing

You may have noticed that when you ask it to print numbers on the screen, the BBC computer usually leaves spaces before the digits.

Type in

```
PRINT 76        [RETURN]
```

and you will see that the micro does not print 76 at the beginning of the line, but about a quarter of the way across the screen. There is a good reason for this, as you will see if you type in this next program:

```
10 PRINT 7
20 PRINT 987132408
30 PRINT 529
40 PRINT 107034
```

RUN it, and you will see that the computer has lined up the numbers so that all the units are in the same column as each other. So are the tens, and the hundreds, etc.

```
          7
987132408
      529
   107034
```

Sometimes you want your numbers printed without spaces. Type in:

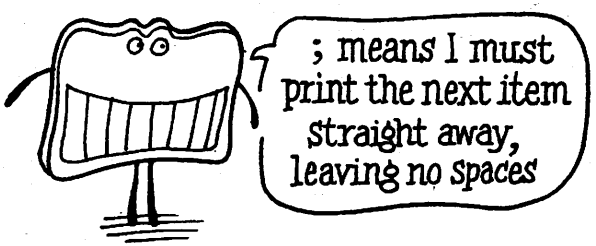
```
PRINT 7 "*" 8          [RETURN]
```

You will see that the spaces make it difficult to read.

To tell the computer "leave no spaces" we use semi-colons.

Try

```
PRINT 7;"*";8          [RETURN]
```



The same thing happens with numbers the computer finds in number variable boxes. Type in this short program:

```
10 INPUT N
20 PRINT "YOUR NUMBER IS" N
```

RUN the program. When you see the ? prompting you, type 100. The computer will print:

```
YOUR NUMBER IS        100
```

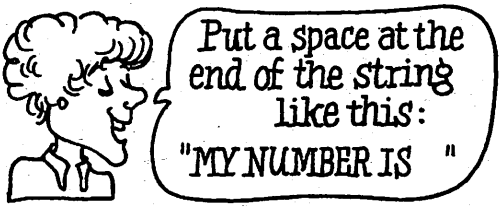
To get rid of the spaces before 100, type line 20 again, so that it reads:

```
20 PRINT "YOUR NUMBER IS";N
```

Now when you RUN the program you have

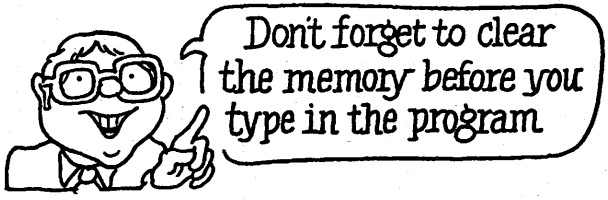
```
YOUR NUMBER IS100
```

How can we get just one space after the word NUMBER?



Another useful trick for making the computer print neatly is to use the comma. Type in this program:

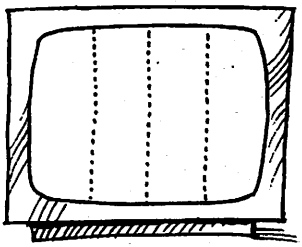
```
  10 CLS
  20 PRINT "JANUARY","FEBRUARY","MARCH"
,"APRIL","MAY","JUNE","JULY"
```



When you RUN it, you will find the names of the months are in neat columns.

```
JANUARY     FEBRUARY    MARCH       APRIL
MAY         JUNE        JULY
```

The screen display of the BBC Micro is divided into four parts, called 'fields'.



A comma tells the computer to move to the beginning of the next field before printing the next item. What do you think would happen if we put in *two* commas between each month in line 20?

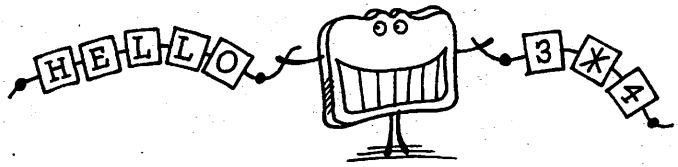
```
20 PRINT "JANUARY",,"FEBRUARY",,"MARCH",,
```

etc.

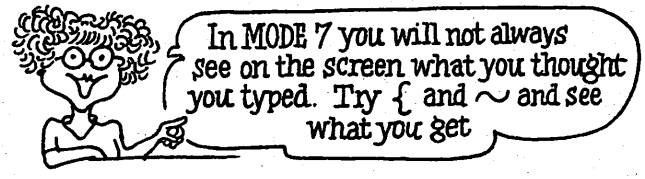
CHAPTER 13

Strings and characters

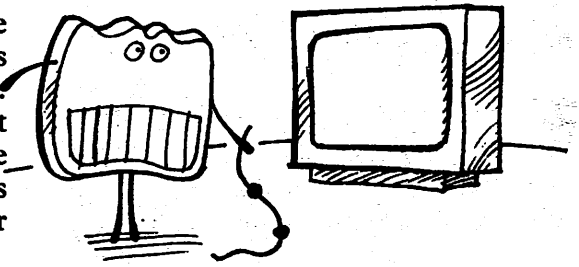
In the programs you have been writing, you have used quotation marks to show the computer that you want something printed on the screen, exactly as you typed it. These items are called 'strings'. You can think of the quotation marks as the knots in the ends of the string.



Strings are made up of 'characters', which are letters, numbers, or any of the other shapes (such as * or !) that you can type in at the keyboard.



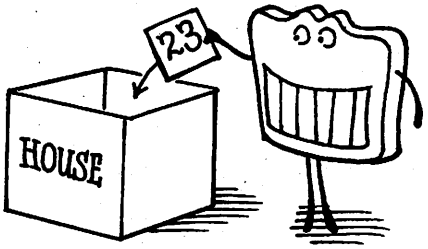
A string can have up to 255 characters in it, or none at all. One with nothing at all between the quotation marks is called an 'empty' or 'null' string.



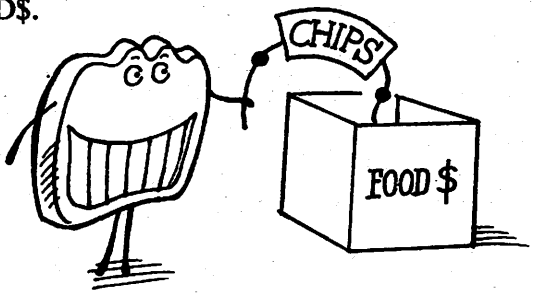
CHAPTER 14

Boxes for strings

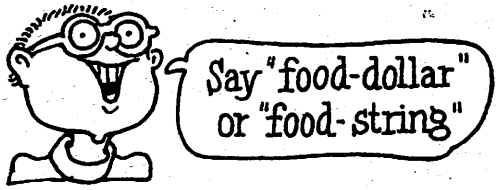
In Chapter 11 we saw that the computer has 'boxes' in its memory for storing numbers in.



It also has boxes, called 'string variables', for storing words, or 'strings' in. In the next program it puts a word into a box called FOOD$.

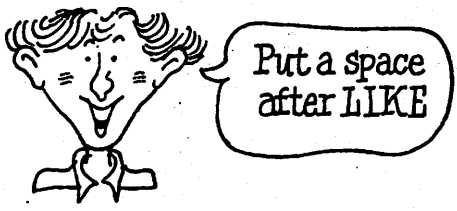


The $ of FOOD$ is very important. It shows that the box is for strings, not numbers.



Here is the program:

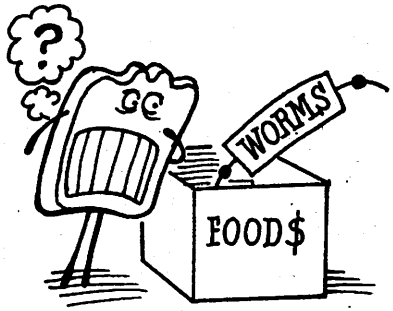
```
10 CLS
20 PRINT "WHAT SORT OF FOOD DO YOU
LIKE BEST?"
30 INPUT FOOD$
40 PRINT "I ALSO LIKE " FOOD$
```



As line 20 is fairly long, it will take up more than one line on the screen. Do not press RETURN when the cursor reaches the right-hand edge of the screen. Just keep typing and it will move down on its own. You will need to press RETURN before you start typing line 30, though.

RUN the program. When you are asked what your favourite food is, type the word and press RETURN. The computer will label a box FOOD$ and put your word in.

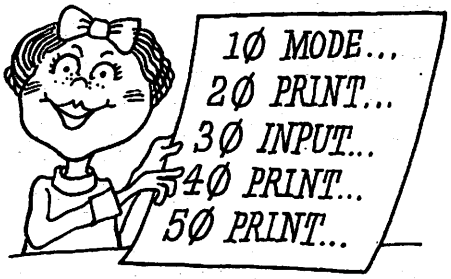
When it comes to line 40, it first prints the words you have put inside the quotation marks (I ALSO LIKE). Then it looks into the box FOOD$ and prints what it finds.



For you to do

Write a program (in MODE 6), in which the computer asks

you which is your favourite animal. You type your answer in, and it goes into as box called ANIMAL$. The computer tells you that it does not like the animal you have chosen. It prefers rams. Make sure that the name of your animal appears in the computer's reply.

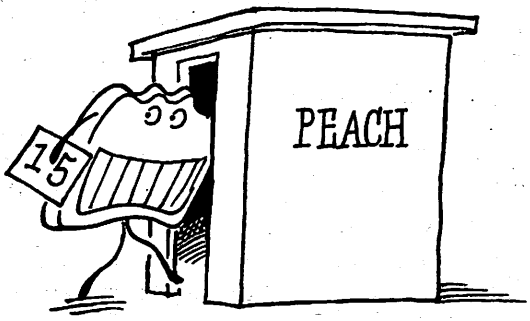


CHAPTER 15

Another way into the boxes

There is another way of putting numbers and strings into variable boxes. Suppose you wanted the computer to store the price of a peach. You could do it like this:

```
LET PEACH = 15
```



Type this in and press RETURN. Now type

```
PRINT PEACH          [RETURN]
```

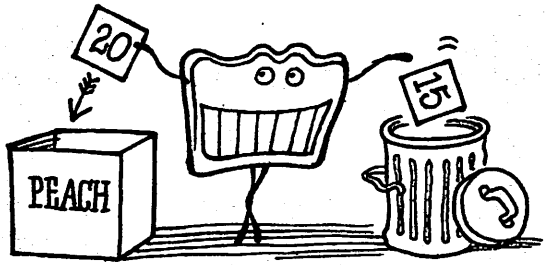
and you will see that the number 15 is stored away in a number variable called PEACH.

Suppose the price of a peach has gone up by 5 pence. We will tell the computer to increase the number in PEACH by 5.

Type in

```
LET PEACH = PEACH + 5          [RETURN]
```

Ask the micro to PRINT PEACH now, and see what answer you get.



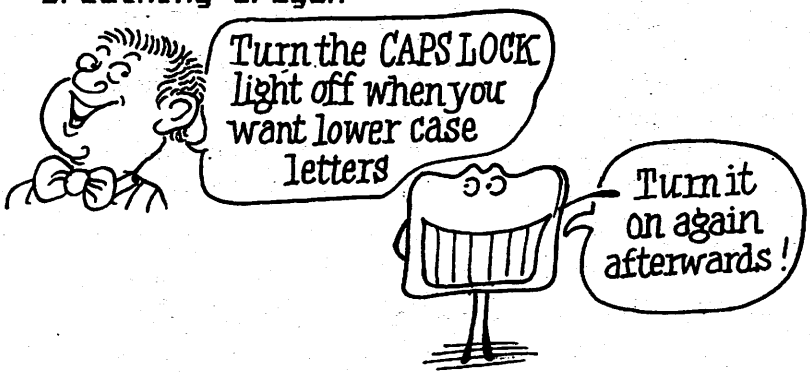
The computer can do all types of mathematics on the numbers in number variable boxes. If the price of a peach went down by 2p, you would tell the computer

```
LET PEACH = PEACH - 2
```

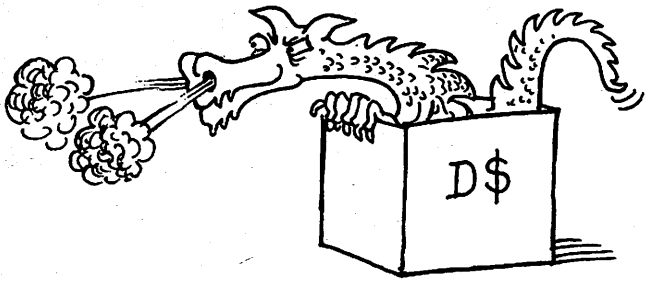
You can use LET to put words into string variable boxes, too.

Suppose you were writing a program which involved printing "the green, scaly, fire-breathing dragon" on the screen a number of times. At the beginning of the program you could put

```
10 LET D$ = "the green, scaly, fire-
breathing dragon"
```



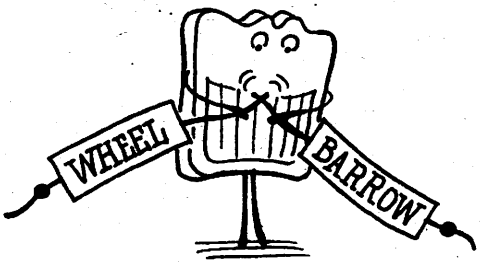
As soon as you RUN the program, "the green, scaly, fire-breathing dragon" is stored in a box called D$. Whenever you need it on the screen, you just tell the computer to PRINT D$ instead of having to type the whole thing again.



It may surprise you to know that the computer can add words together, using the + sign. Tell it to

```
LET W$ = "WHEEL" + "BARROW"
```

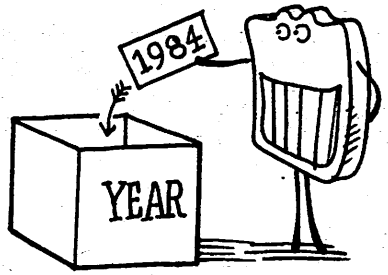
and then find out what is in the box W$.



For you to do

Write a program which:

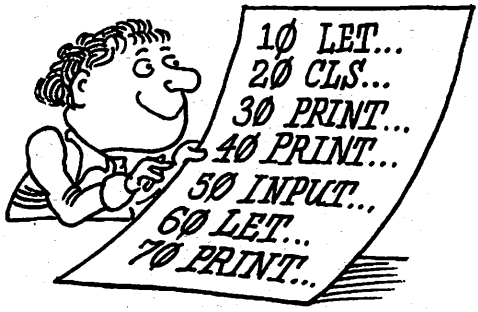
The computer puts the number for this year into a box called YEAR.



It clears the screen, and asks you to type in your age on your birthday this year. Your answer goes into AGE.

Then it words out when you were born, putting the year into a box called BIRTH.

Lastly it tells you when you were born.

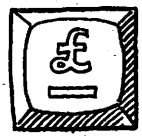


CHAPTER 16

Names for your variable boxes

There are a few rules to remember when choosing names for your number and string variable boxes.

1. The name must start with a letter, which can be a capital or in lower case. You may have numbers in a variable name, but not right at the beginning. You can have a box called A2, but not 2A.
2. There must be no spaces in the middle of the name.
3. Apart from letters and numbers, there is one other character you can use. It is the line on the same key as the pound sign.

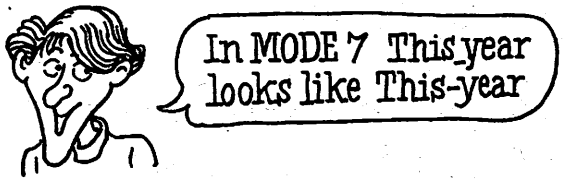


This is a useful character to use if you really wanted to put in a space. Although the computer would not be able to call a box

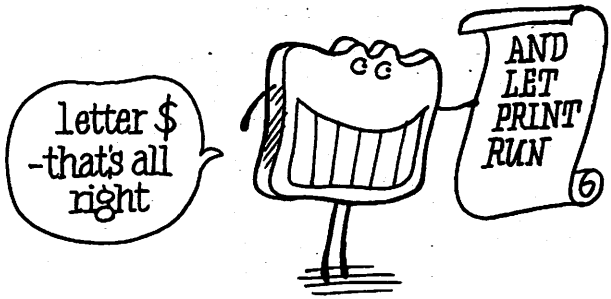
```
This year
```

it could call one

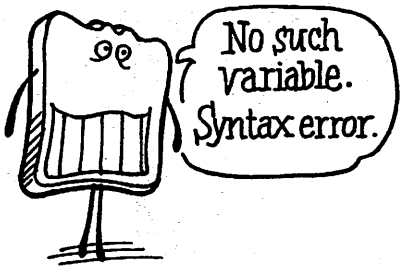
```
This_year
```



4. A variable name must not start with a BASIC keyword. You could not call one LETTER$, because it begins with LET. You could call it letter$, though, because the micro only recognises BASIC keywords if they are in capitals.



Because you do not yet know all the BASIC keywords, you may start a variable name with one by mistake. If you do, the computer will let you know, when you RUN the program.



There is a BASIC word 'TO' which you will meet later in this book. It is easy to forget this, and try to call a variable TOTAL or TOMORROW.

For you to do

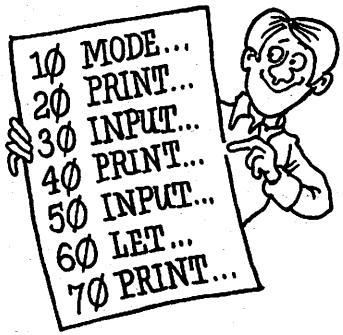
Write a program in MODE 7 in which:

The computer first asks you for a number. You type one in, and it goes into a number variable box.

Then the computer asks for another number, which is stored in another number variable box.

The computer adds together your two numbers and puts the answer into a third box.

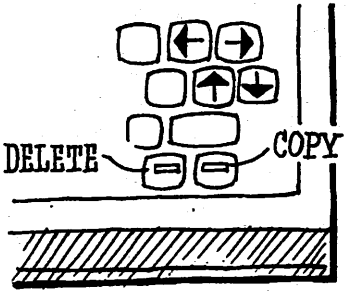
It prints the answer, together with a suitable message.



CHAPTER 17

Making Changes

One way of changing a line of your program is to type the whole line again. There is another way. On the right side of the keyboard are five keys which are lighter in colour than the others. These keys, four arrows and COPY, are for 'editing' or altering program lines which are on the screen. DELETE, which is next to COPY, is also useful in editing.



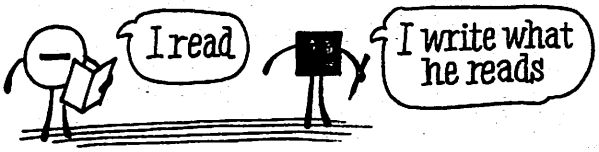
Before you can try the editing keys, you will need a program in the memory. Type this one in, and then we'll make some changes:

```
10 MODE 6
20 LET NAME$ = "BERT"
30 PRINT "WHAT IS MY NAME?"
40 INPUT GUESS$
50 PRINT "YOU GUESSED " GUESS$
60 PRINT "MY NAME IS " NAME$
```

Let's change the program, so that it runs in MODE 4,

instead of MODE 6. Press the key with the arrow pointing upwards.

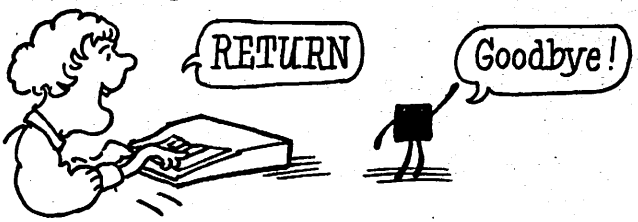
As soon as you do this, you are in 'editing mode', and you have two cursors on the screen. The normal cursor (the short flashing line) is now called the 'read cursor'. The other is a solid rectangular block — the 'write cursor'.



Using the four arrows, you can move the read cursor anywhere on the screen. Try it. Now move the flashing line until it is immediately below the 1 of line 10. Press COPY lightly. You will find that both cursors have moved one place to the right, and that the 'write cursor' has written the number 1. Keep pressing COPY until you have

```
10 MODE ■
```

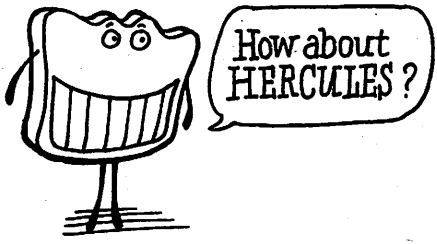
on the bottom line of the screen. If you go too fast, and COPY more than you need, you can use DELETE to remove any unwanted characters from your new line. Now type 4 (as we wish to change to MODE 4) and press RETURN. As soon as you press RETURN, your new line is in the memory, and you are out of the editing mode.



Next we'll change the name which is to be stored in NAME$. Using the arrow keys, move the flashing 'read cursor' until it is under the 2 of line 20. Press COPY until the 'write cursor' has written

```
20 LET NAME$ = "
```

Decide what name you are going to have, and type it in.



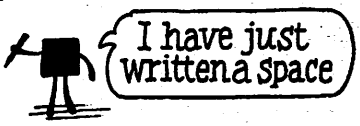
Put in some quotation marks and press RETURN.

Lastly, let's change line 60 so that it reads

```
60 PRINT "MY NAME IS REALLY " NAME$
```

Press the upwards arrow until the read cursor is under the 6 of line 60. Press COPY until you have

```
60 PRINT "MY NAME IS
```



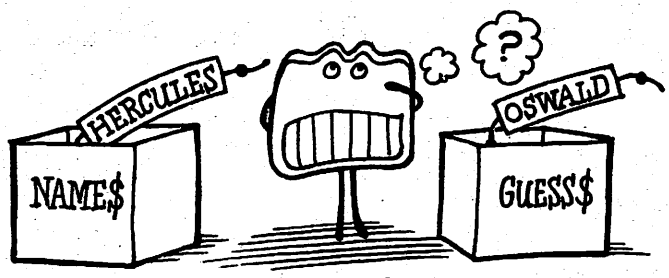
on the bottom line of the screen. Type the word REALLY and a space. Now move the read cursor until it is under the second set of quotation marks:

```
60 PRINT "MY NAME IS " NAME$
```

Now press COPY until the write cursor has written

```
60 PRINT "MY NAME IS REALLY " NAME$
```

Press RETURN and RUN the program.



CHAPTER 18

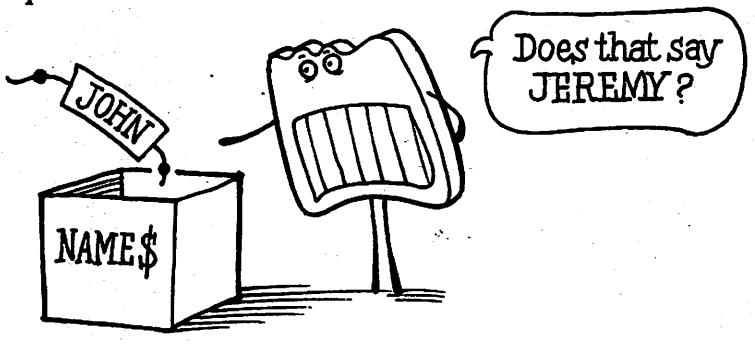
A decision for your Micro

Here is a program in which your BBC Micro is rather choosy about who it speaks to:

```
10 CLS
20 PRINT "WHO ARE YOU?"
30 INPUT NAME$
40 IF NAME$ = "JEREMY" THEN PRINT "GR
EETINGS " NAME$
```

When you are typing this program in, put in your own name, instead of JEREMY.

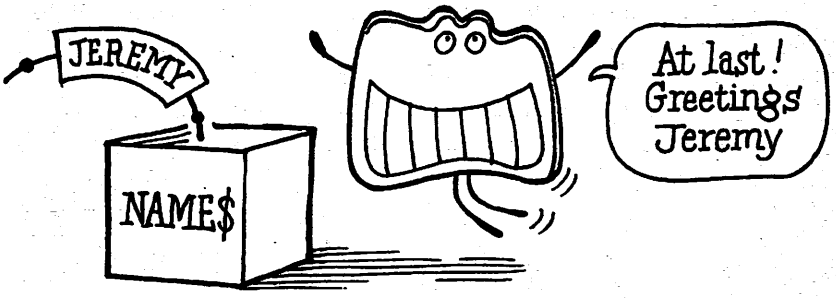
RUN the program and when asked, type in a name. It will go into the string variable NAME$. When the computer comes to line 40 it has to look in the box NAME$ and ask itself a question:



IF the name it is looking for is in the box, THEN it prints its greetings on the screen. Otherwise, it does nothing.

RUN the program a few times, putting in different names

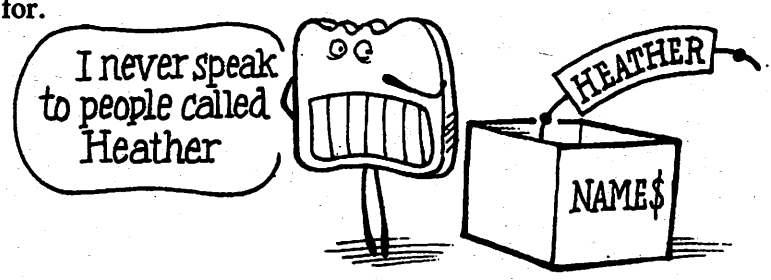
when asked, and see what happens.



Now tell the computer to LIST the program, and use the editing keys (the arrows and COPY) to add some more words to line 40:

```
10 CLS
20 PRINT "WHO ARE YOU?"
30 INPUT NAME$
40 IF NAME$ = "JEREMY" THEN PRINT
"GREETINGS " NAME$ ELSE PRINT "I NEVER
SPEAK TO PEOPLE CALLED " NAME$
```

The word ELSE introduces the instruction which tells the computer what to do if it does not find the name it is looking for.



For you to do

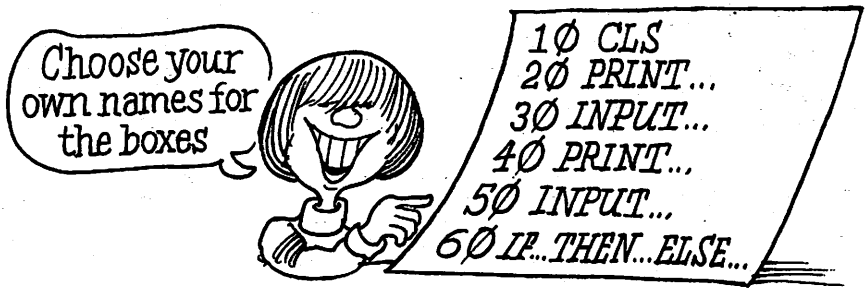
Write a program in which the computer first clears the screen and then asks you your name.

When you type it in, it is stored in a string variable.

Then the computer asks you whether you think computers are clever.

Your answer goes into another string variable.

If you answered "YES" the computer is delighted. It tells you how marvellous you are (calling you by name). If you didn't answer "YES", the computer is disgusted, and says so.



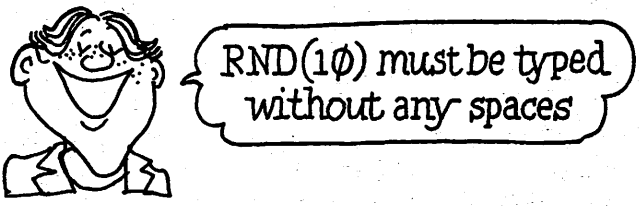
CHAPTER 19

Think of a number

In many games, it is necessary for the computer to choose a number, without the human knowing what it is. The BASIC keyword which tells your BBC Micro to do this is RND.

If you want the computer to choose a number between 1 and 10, and print it on the screen, you type

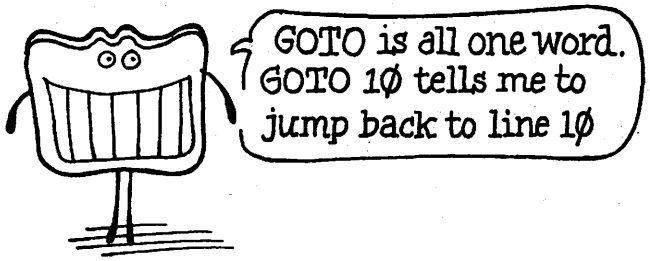
```
PRINT RND(10)        [RETURN]
```



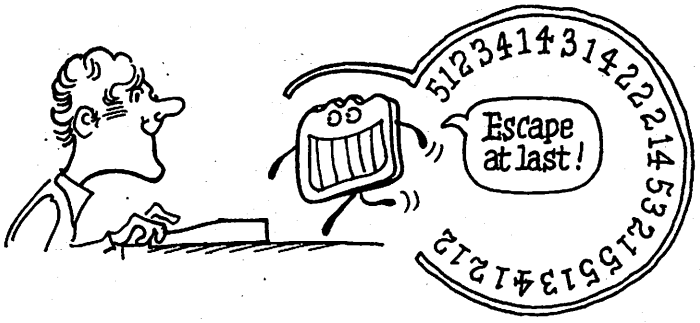
To change the range of numbers the computer has to choose from, change the number in the brackets.

Here is a program in which the computer keeps choosing numbers between 1 and 5:

```
10 PRINT RND(5)
20 GOTO 10
```



When you RUN this program you will see numbers flashing past your eyes up the screen. This is because you have put your micro into a never-ending loop. Whenever it gets to line 20, it is told to go back to line 10 again. The only way out of the loop is to press the ESCAPE key.

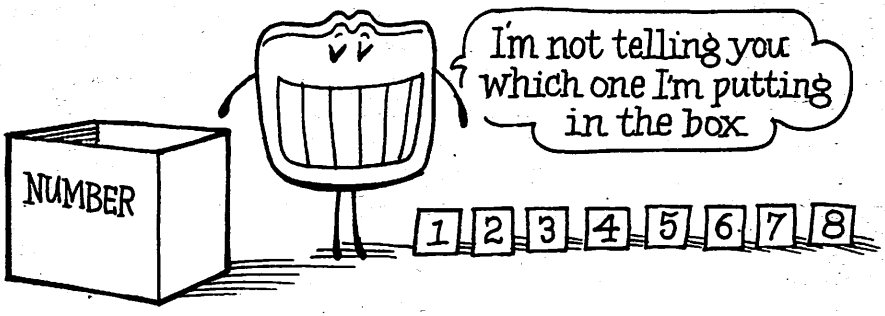


You can use the ESCAPE key whenever you want to stop the running of a program. It is different from BREAK, because it does not wipe your program from the memory. Type LIST and press RETURN, and you will see your program on the screen again.

Look at the numbers on the screen and you will see that the smallest is 1 and the largest is 5. There is no pattern to the numbers; the computer chose them at random.

In this next program, you have to guess the number the computer has chosen:

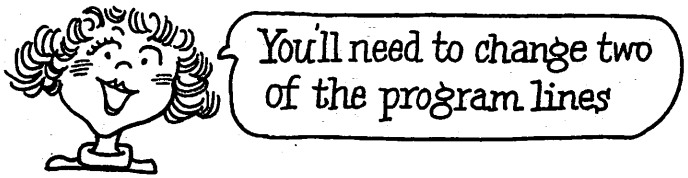
```
10 CLS
20 LET NUMBER = RND(8)
30 PRINT "MY NUMBER IS BETWEEN 1 AND
8"
40 PRINT "TYPE IN YOUR GUESS"
50 INPUT GUESS
60 IF GUESS = NUMBER THEN PRINT
"RIGHT" ELSE PRINT "WRONG"
```



RUN the program and see if you can guess the number. You only get one chance!

For you to do

Change the program so that the computer chooses a number between 1 and 12.



Now put in another line (line 70), so that the computer tells you what the correct number was.

CHAPTER 20

The computer does it again

One of the very useful things about computers is that they can carry out the same instructions over and over again, until you tell them to stop. In Chapter 19, we wrote a program in which you had to guess the number the computer had chosen. Unfortunately for you, you didn't have much chance of getting it right, as you were only allowed one guess. It would be fairer on you (though perhaps not on the computer!) if it kept asking you to type in a guess, until you got it right.

For this we can use the BASIC key words REPEAT and UNTIL.

First of all we have to decide which particular part of the program has to be repeated. (If you do not have the number guessing program in the computer's memory at the moment, it would be a good idea to type it in now, so that it is ready to change.)

REPEAT goes at the beginning of the part to be repeated, and UNTIL goes at the end.

We do not want the computer to keep choosing a new

number (line 20), nor do we need it to keep printing "MY NUMBER IS BETWEEN 1 AND 8" (line 30). Here is the part we want the computer to repeat:

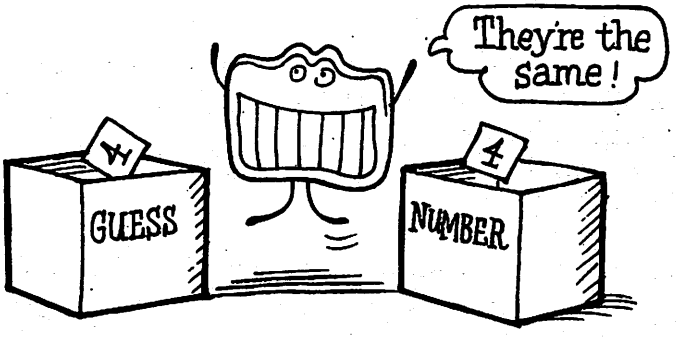
```
40 PRINT "TYPE IN YOUR GUESS"
50 INPUT GUESS
60 IF GUESS = NUMBER THEN PRINT
"RIGHT" ELSE PRINT "WRONG"
```

Before line 40 must go the word REPEAT, so we put in

```
35 REPEAT
```

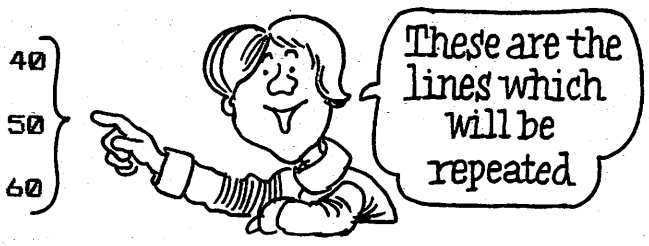
These lines need to be repeated until you have guessed the correct number so, after line 60 we put in

```
70 UNTIL GUESS = NUMBER
```



LIST the program, and it should now look like this:

```
   10 CLS
   20 LET NUMBER = RND(8)
   30 PRINT "MY NUMBER IS BETWEEN 1 AND
8"
   35 REPEAT
   40 PRINT "TYPE IN YOUR GUESS"
   50 INPUT GUESS
   60 IF GUESS = NUMBER THEN PRINT
"RIGHT" ELSE PRINT "WRONG"
   70 UNTIL GUESS = NUMBER
```



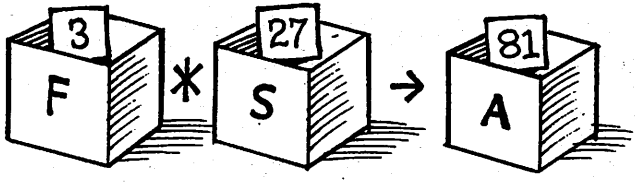
See how many tries you need before you guess the number.

How about making the computer congratulate you when you guess right? Type in

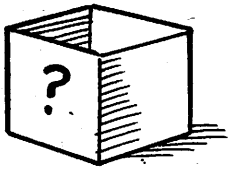
This message will not be printed until you have guessed right, as the computer has to keep going round and round lines 40 to 60 until you are correct.

For you to do

In your next program the computer asks some very difficult multiplication questions. First it chooses a number between 1 and 100 and stores it in a variable box. Then it chooses another number between 1 and 100 and puts that in another variable box. It multiplies the two numbers together and the answer goes into a third box.



Now it asks you to multiply its two numbers together. You type in what you think is the answer (where does your number go in?)



The computer repeats its question until you give the correct answer. Then it prints a "well done" message.

CHAPTER 21

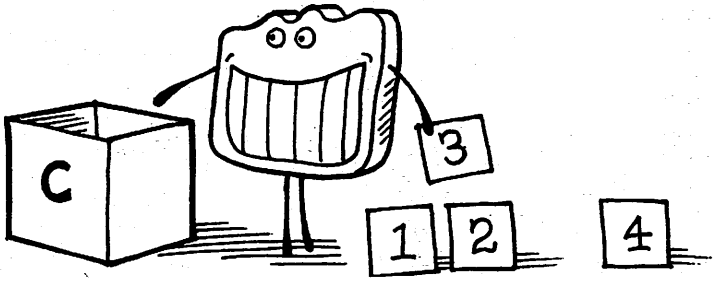
Think of a word

In this program, the computer chooses a colour, and asks you to guess it:

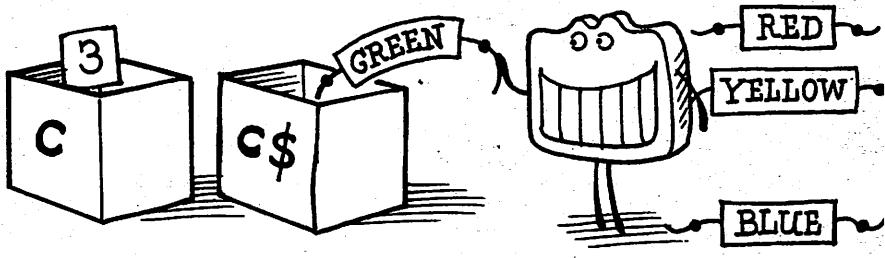
```
 10 CLS
 20 LET C = RND(4)
 30 IF C = 1 THEN LET C$ = "RED"
 40 IF C = 2 THEN LET C$ = "YELLOW"
 50 IF C = 3 THEN LET C$ = "GREEN"
 60 IF C = 4 THEN LET C$ = "BLUE"
 70 PRINT "WHAT COLOUR AM I THINKING
OF?"
 80 REPEAT
 90 INPUT G$
100 IF G$ <> C$ THEN PRINT "NO"
110 UNTIL G$ = C$
120 PRINT "YES"
```

We use three variable boxes in this program.

C is a number variable, and the random number which is chosen at line 20 goes in here.



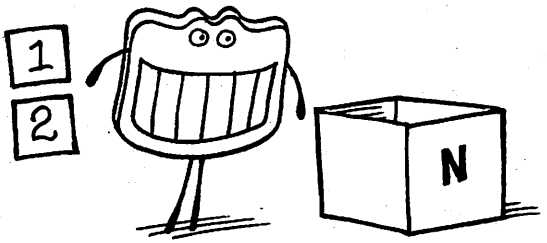
C$ is a string variable. The colour the computer is thinking of is stored in it. Which colour it is depends on what number is in C.



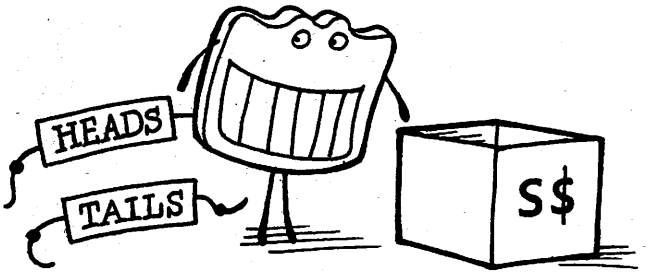
G$ holds your guess, which you type in at line 100.

For you to do

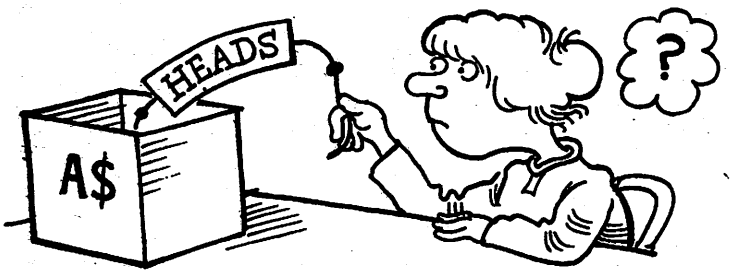
Write a program in which the computer first clears the screen. Then it chooses a number either 1 or 2, and puts it into a number variable box called N.



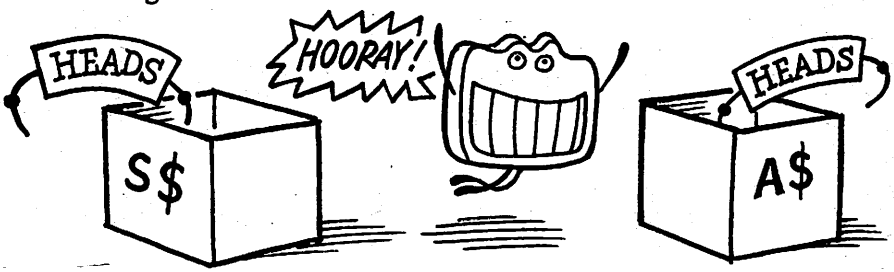
According to whether it chose 1 or 2, the word HEADS, or the word TAILS, goes into a string variable box called S$.



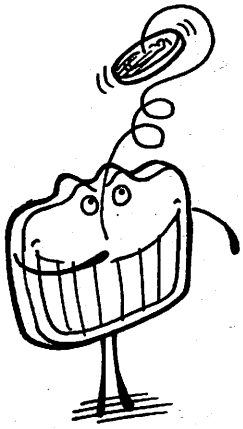
The computer asks you to guess HEADS or TAILS. You'll need another string variable box for your answer, call it A$.



The computer checks your answer and tells you whether you were right.



Now put in a REPEAT . . . UNTIL loop, to make the computer keep repeating the whole program, except the CLS at the beginning, until it finds the word STOP in A$.

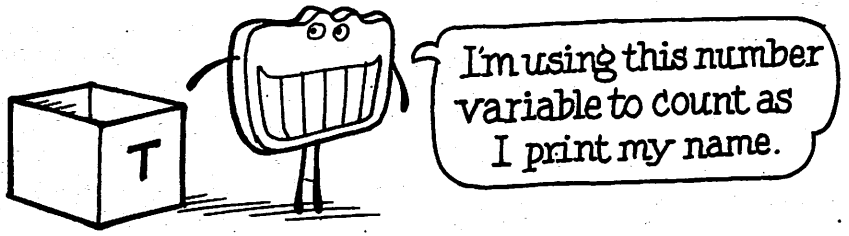


CHAPTER 22

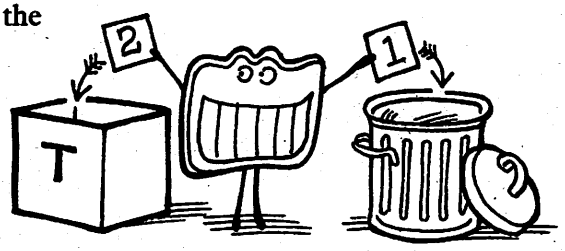
Loops for counting

Here is a program in which the BBC Micro prints its name fifteen times:

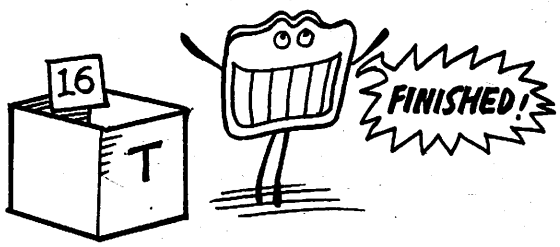
```
10 CLS
20 LET T = 1
30 REPEAT
40 PRINT T;"BBC MICRO"
50 LET T = T + 1
60 UNTIL T > 15
70 PRINT "FINISHED"
```



The box T starts off with the number 1 in it, ready for the first printing. At line 40 the computer looks in T and prints the number it finds. Then it prints "BBC MICRO". Line 50 tells it to add 1 to the number in T, ready for the next printing.



At line 60 it checks to see what number is in T. If it is 15 or less, it goes back to print its name again. If it is more than 15, it knows it has finished the loop, and moves on to line 70.

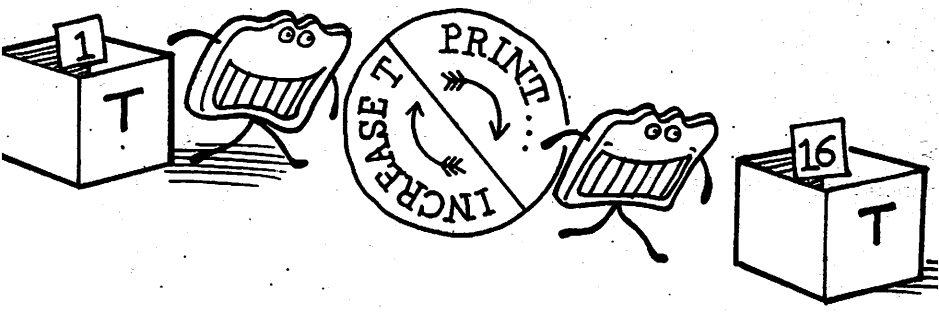


Now here is a shorter way to do the same thing, using another kind of loop, which is specially for counting. It is called a FOR . . . NEXT loop.

```
10 CLS
20 FOR T = 1 TO 15
30 PRINT T;"BBC MICRO"
40 NEXT T
50 PRINT "FINISHED"
```

When you RUN this program, you will see that what comes up on the screen is just the same as in the last program.

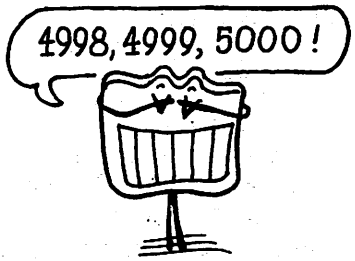
Lines 20, 30 and 40 are the "FOR . . . NEXT" loop. Line 20 tells the computer to use box T for counting, and to start at 1 and finish at 15. Line 40 increases the number in T and if necessary sends the micro back to print its name again.



FOR . . . NEXT loops are very useful when you want to make the computer pause in the middle of a program.

```
10 PRINT "START"
20 FOR A = 1 TO 5000
30 NEXT A
40 PRINT "STOP"
```

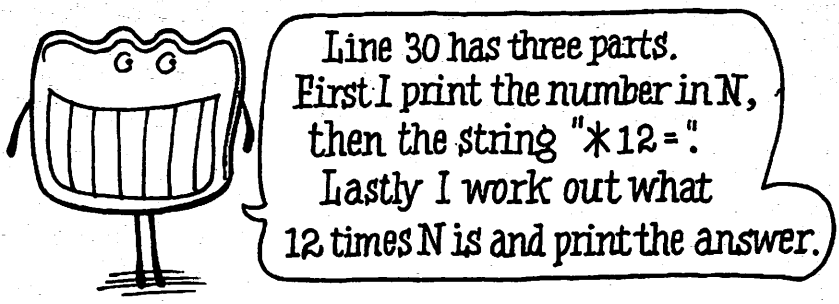
In this program, the micro prints START, counts to 5000, and then prints STOP. It won't print anything while it is counting, as there is no PRINT instruction inside the loop.



RUN the program. You will probably be surprised at how fast the BBC Micro counts.

Here is a program to print out the 12 times table, up to 10 times 12:

```
10 CLS
20 FOR N = 1 TO 10
30 PRINT N; "*12="; N*12
40 NEXT N
```


Line 30 has three parts.
First I print the number in N,
then the string "*12=".
Lastly I work out what
12 times N is and print the answer.

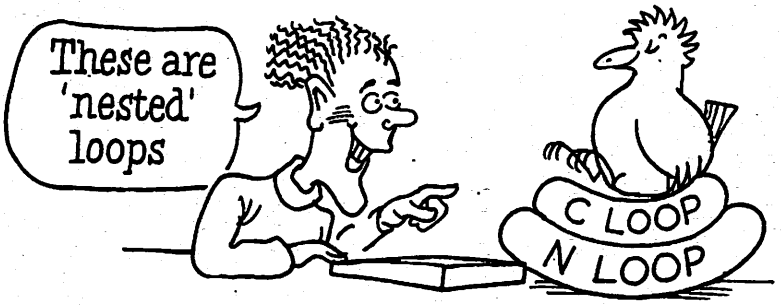
You can put one FOR . . . NEXT loop *inside* another. Type in these extra lines:

```
33 FOR C = 1 TO 2000
37 NEXT C
```

The program now looks like this

```
10 CLS
20 FOR N = 1 TO 10
30 PRINT N; "*12="; N*12
33 FOR C = 1 TO 2000
37 NEXT C
40 NEXT N
```

Now the computer will pause and count to 2000 after printing each line.



For you to do

Write a program in which the computer prints the 16 times table, up to 20 times 16. Tell it to pause and count to 3000 after printing each line.

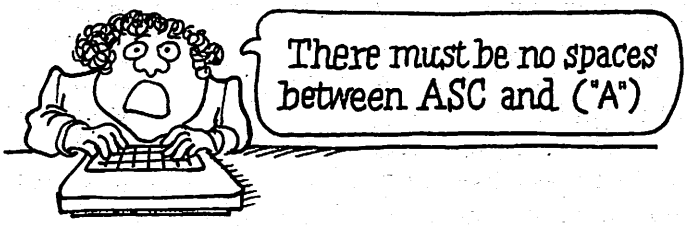
CHAPTER 23

Characters and codes

Each of the characters the BBC Micro can print on the screen has a number which goes with it, called its ASCII CODE number.

The character "A", for example, has ASCII code number 65. To get the computer to tell you this, type in

```
PRINT ASC("A")          [RETURN]
```

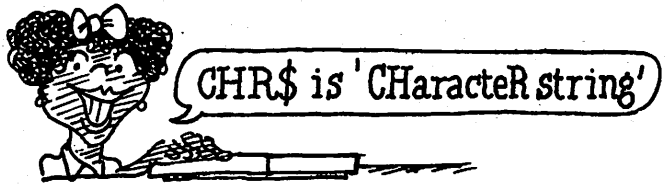


Now see if you can find out the code for "B".

If you give the computer a code number, it can tell you which character it belongs to.

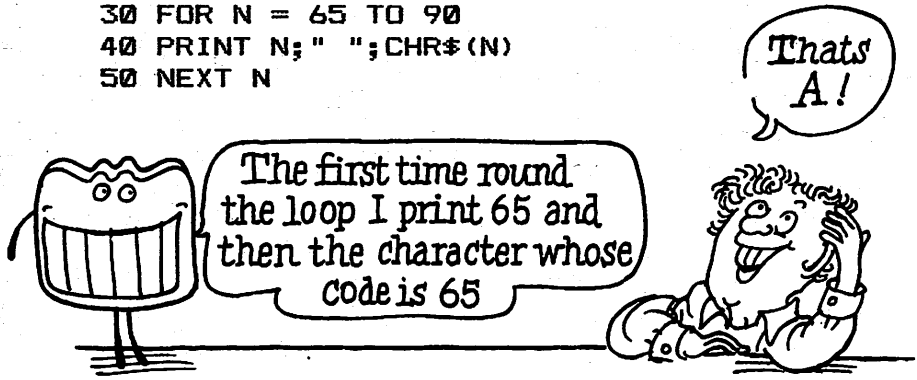
```
PRINT CHR$(98)          [RETURN]
```

will tell you which character has number 98 for its code.



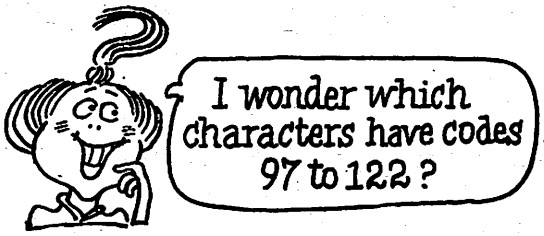
The characters on the keyboard have codes 32 to 126. Here is a program which prints the capital letters with their ASCII code numbers.

```
10 MODE 4
20 PRINT "ASCII CODE  CHARACTER"
30 FOR N = 65 TO 90
40 PRINT N;"  ";CHR$(N)
50 NEXT N
```



The reason it is in MODE 4 rather than MODE 7, is that in MODE 7 you can only have 25 lines of print on the screen at a time, which is not enough for the 26 letters of the alphabet. In MODE 4 you can have 32 lines. If you look at page 170 you will find details of the eight MODES.

By changing the numbers in the "FOR . . . TO" instruction (line 20) you can make the computer show you the characters which have other ASCII codes.



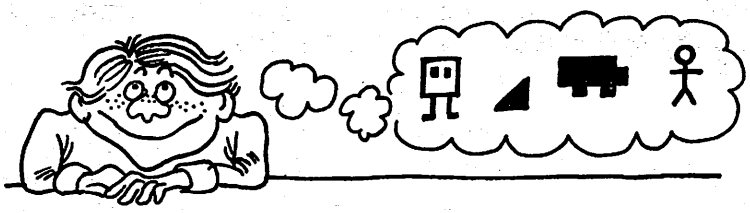
For you to do

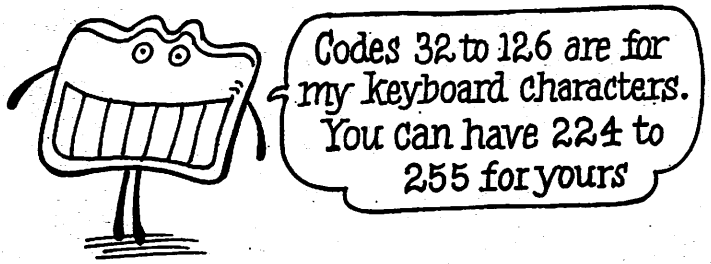
Do you remember in chapter 9 (MODES) you found that in MODE 7, some of the keys gave you unexpected characters on the screen? Ask the computer to show you the characters 123 to 126, first in MODE 4 and then in MODE 7.

CHAPTER 24

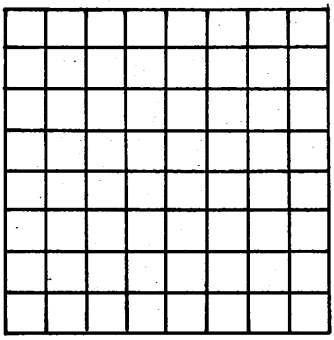
Your own characters

If you want to write a games program, with cars or invaders moving about the screen, you are going to need some different characters from those you find on the keys. Luckily the BBC Micro has some spare ASCII code numbers, and you can design your own characters for them. The code numbers available for you are numbers 224 to 255. You can use them only in MODES 0 to 6 – not in MODE 7.

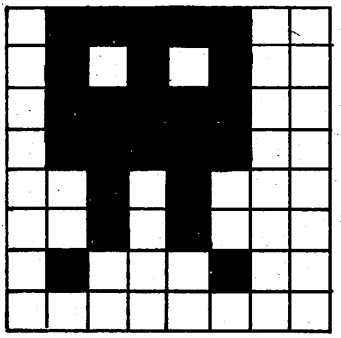




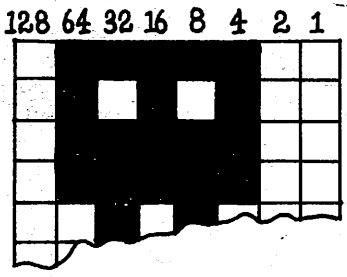
First you need to design your character. You will need squared paper for this. Mark off a section eight squares across by eight squares down.



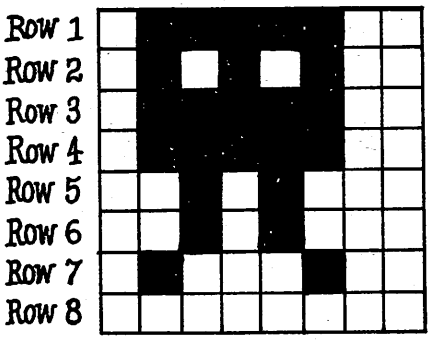
Now draw your character, by shading in some of the squares (which are known as 'pixels'.) Here is an alien invader.



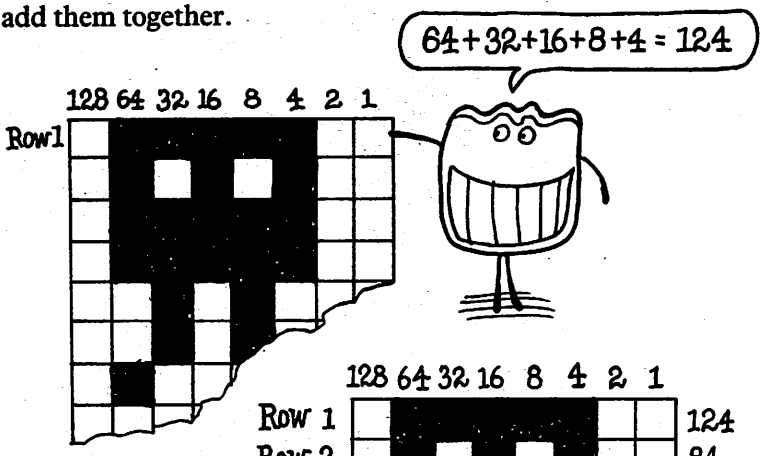
Next you need to number the columns, like this:



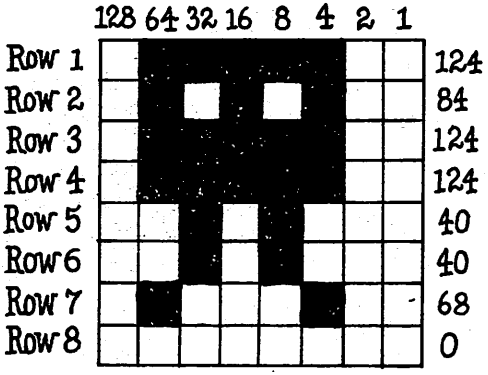
Now number the rows:



Look at row 1, and see which of its squares are shaded in. Now write down the numbers above the shaded squares, and add them together.



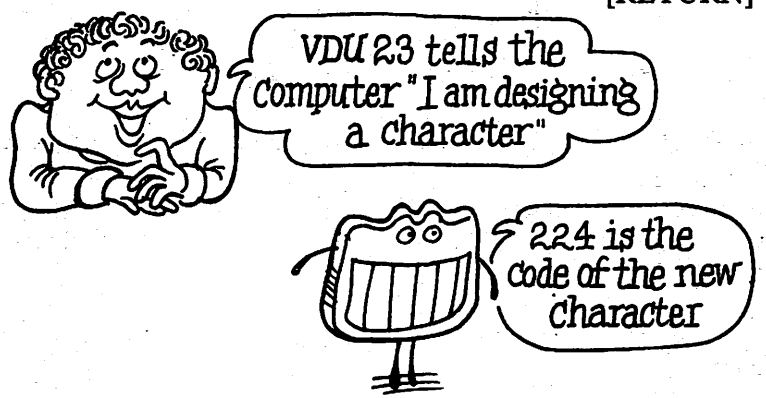
Do the same for the other rows.



These eight numbers, 124, 84, 124, 124, 40, 40, 68, 0, will tell the computer what the invader looks like.

We are ready to instruct the computer to store the invader in its memory. We will give it code number 224. Type in:

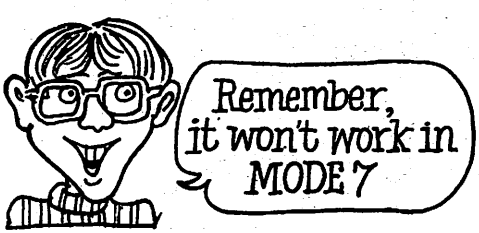
```
VDU 23,224,124,84,124,124,40,40,68,0
                                [RETURN]
```



The last eight numbers of the line are the numbers from the eight rows of the grid.

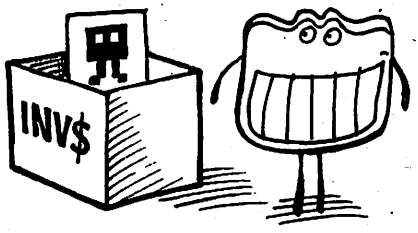
When you want to see your invader, you tell the computer to print the character with code 224:

```
PRINT CHR$(224)     [RETURN]
```



You can tell the computer to store the invader in a string variable box.

```
LET INV$ = CHR$(224)
```



When it sees PRINT INV$, the computer will then print the invader.

Here is a program which designs a small car, gives the new character code number 225, and then prints a screenful of it:

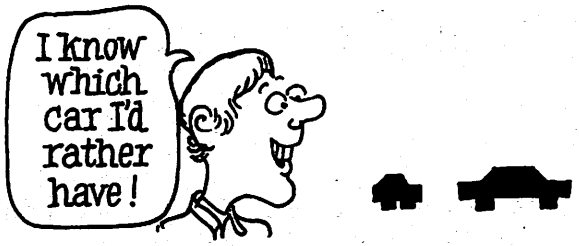
```
   10 MODE 5
   20 VDU 23,225,0,0,48,120,255,255,
102,102
   30 LET CAR$ = CHR$(225)
   40 FOR C = 1 TO 207
   50 PRINT " "; CAR$; " ";
   60 NEXT C
```

Line 50 prints a space, a car, then another space. Do you know why there is a semi-colon at the end of the line? Take it out and RUN the program again and you'll see why.

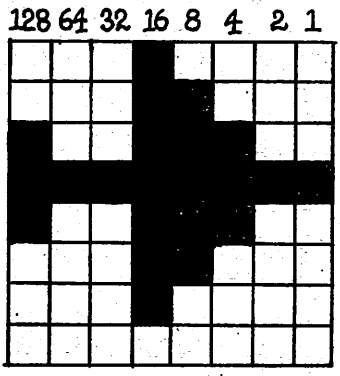
Now put the semi-colon back and change line 10 to

```
10 MODE 4
```

RUN the program and see what difference this makes.



For you to do



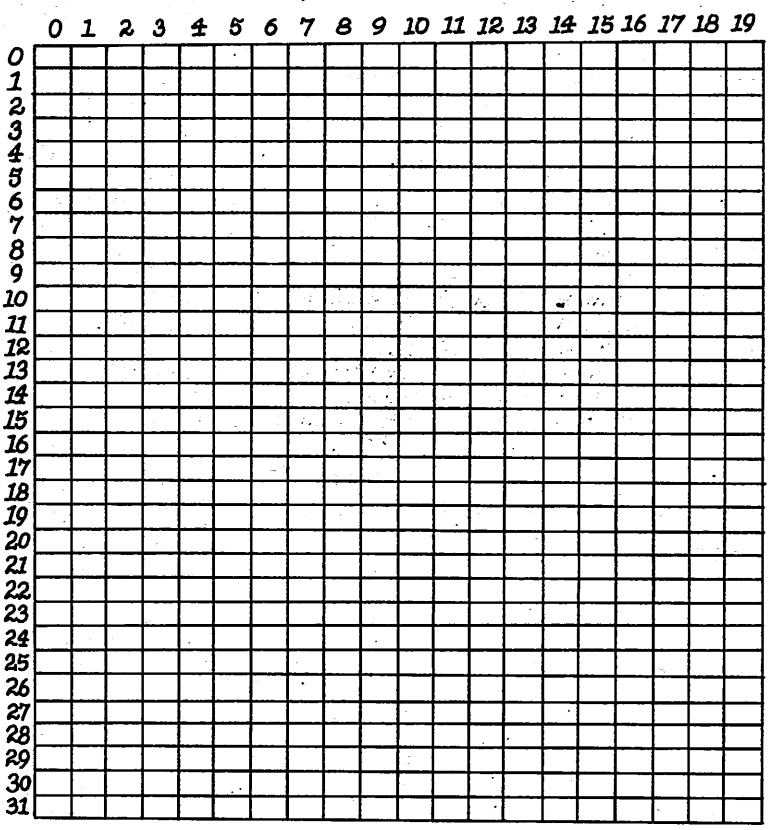
Here is a plane. Work out the eight row numbers. Tell the computer you wish this character to have code number 250. Then get it to print a screenful of planes in MODE 6.

CHAPTER 25

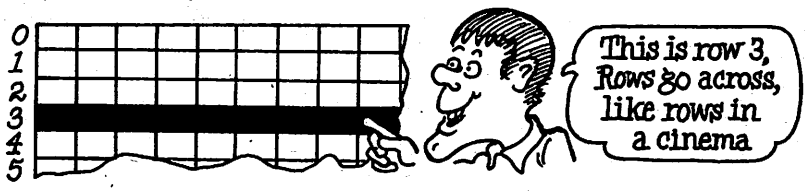
Choose your spot on the screen

So far, after you have told it to clear the screen, your micro has started printing in the top left corner of the screen.

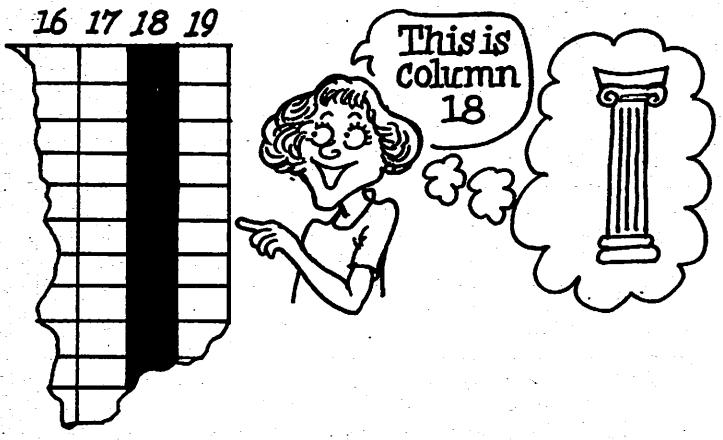
Luckily, there is a way of telling the BBC Micro just where to print. Imagine that your micro is in MODE 5, and that your T.V. screen is ruled into boxes, like this:



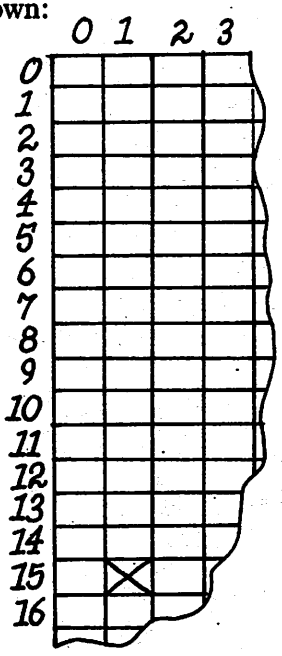
There are 32 rows, which are numbered from 0 (top row), to 31 (bottom row).



There are 20 columns. They go from 0 on the left, to 19 on the right.

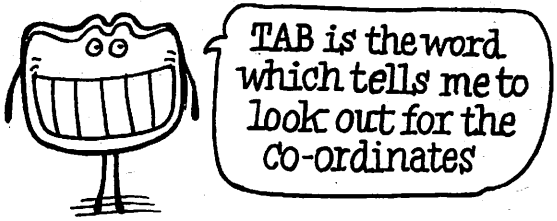


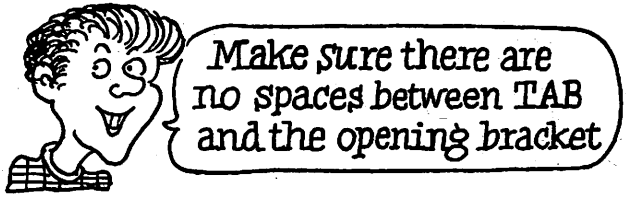
Suppose we want the letter X near the left of the screen, about halfway down:



We have to give the computer first the column number, which is 1, and then the row number, which is 15. These are called the 'co-ordinates'. Change to MODE 5 and type

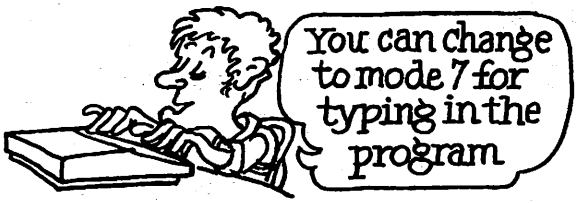
```
PRINT TAB(1,15) "X"        [RETURN]
```





If you want to print a word at a particular place on the screen, you tell the computer which square the first letter is to go in. This program will print the name "ROBERT" near the middle of the screen.

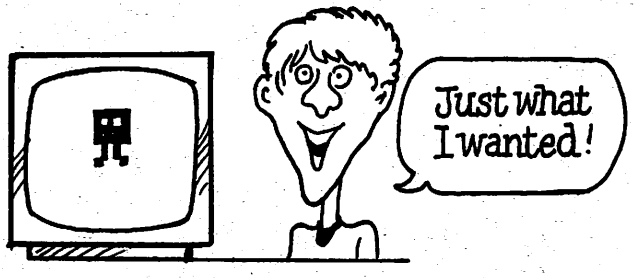
```
10 MODE 5
20 PRINT TAB(8,15) "ROBERT"
```



For you to do

Write a program to design an alien invader. (You can use

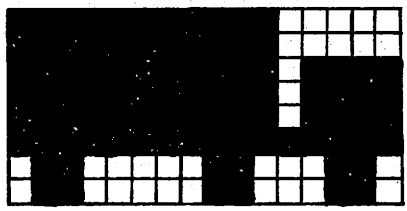
the one from the last chapter if you like). Then tell the computer to print it in the middle of the screen.



CHAPTER 26

Your characters move

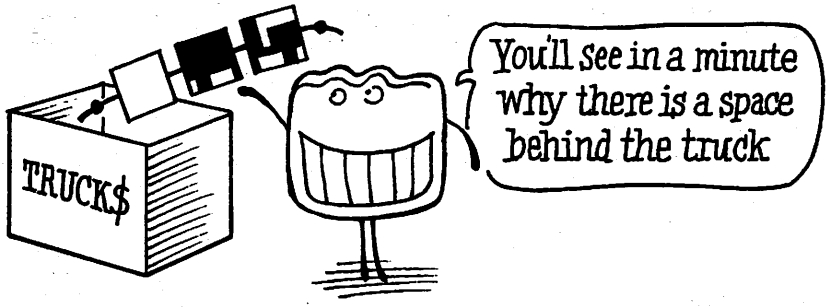
Here is an articulated truck:



Because of its length, it is made up of two characters. The back will have code number 230, and the front code number 231.

In the next program, lines 20 and 30 put the information about these two characters into the computer's memory. At line 40, the computer stores the truck, with a space behind it, in TRUCK$, and at line 60 it prints it.

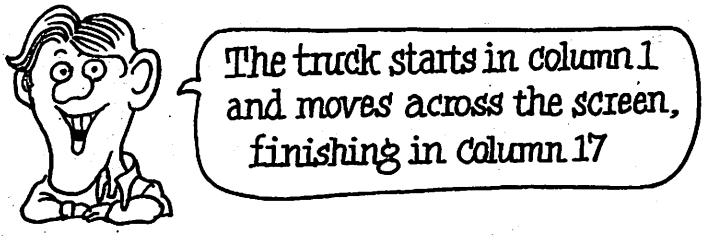
```
   10 MODE 5
   20 VDU 23,230,255,255,255.255,255,
255,96,96
   30 VDU 23.231,224,224,239,239,239,
255,198,198
   40 LET TRUCK$ = " " + CHR$(230) +
CHR$(231)
   60 PRINT TAB(1,15) TRUCK$
```



To make the truck move across the screen, we print it again and again, increasing the column number each time. A FOR . . . NEXT loop is handy here.

Type in

```
50 FOR COLUMN = 1 TO 17
```



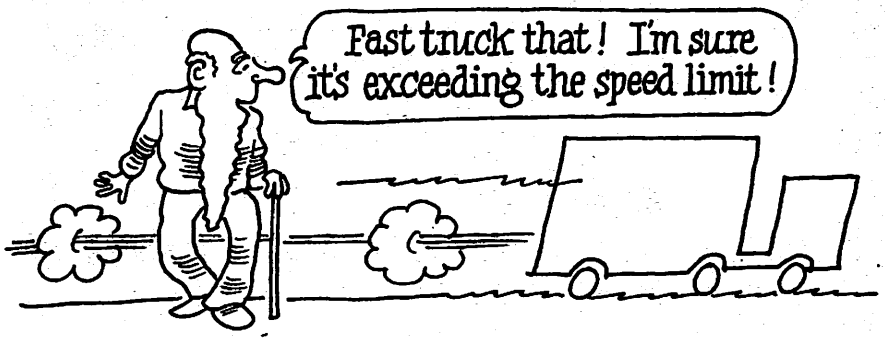
Change line 60 to

```
60 PRINT TAB(COLUMN, 15) TRUCK$
```

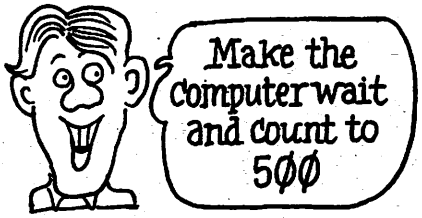
and type in

```
90 NEXT COLUMN
```

Now RUN the program.



How can we slow it down?

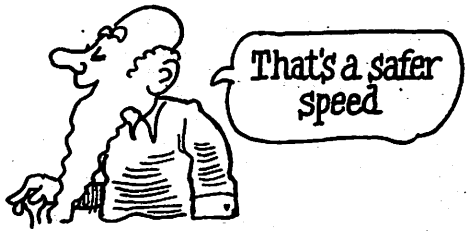


Type in

```
70 FOR W = 1 TO 500
80 NEXT W
```

The whole program now looks like this:

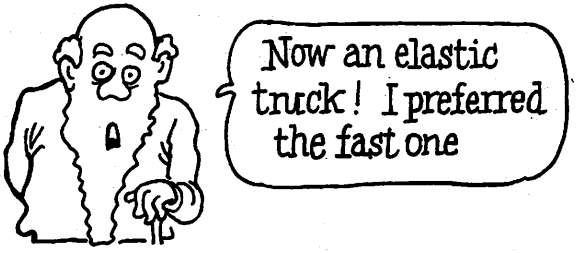
```
   10 MODE 5
   20 VDU 23,230,255,255,255,255,255,
255,96,96
   30 VDU 23,231,224,224,239,239,239,
255,198,198
   40 LET TRUCK$ = " " + CHR$(230) +
CHR$(231)
   50 FOR COLUMN  = 1 TO 17
   60 PRINT TAB(COLUMN, 15)  TRUCK$
   70 FOR W = 1 TO 500
   80 NEXT W
   90 NEXT COLUMN
```



If you are still wondering why we put a space behind the back of the truck, change line 40 to

```
40 LET TRUCK$ = CHR$(230) + CHR$(231)
```

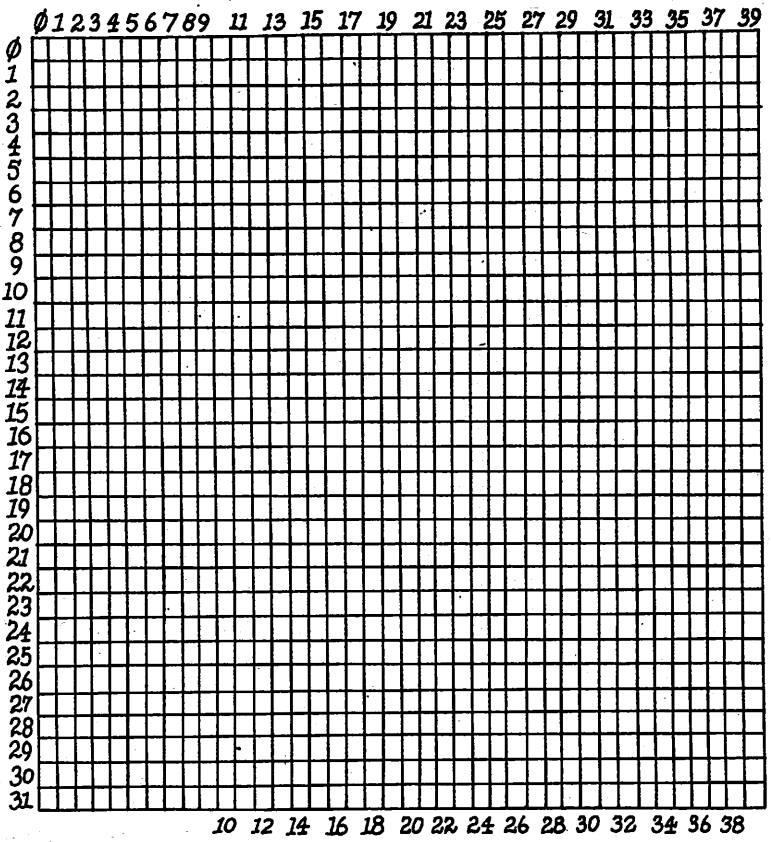
and RUN the program again.



As each new truck is printed, the space is needed to blot out the back of the previous truck.

You will need this truck program again in a later chapter, so you might like to turn to Chapter 32 now and find out how to record it on to a cassette.

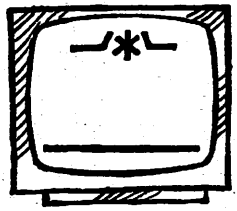
As you have probably discovered, when the BBC Micro is in MODE 4, it can fit 40 characters across the screen. So when you are planning screen displays in this MODE, you need to imagine the T.V. screen ruled up like this.



Here the columns are numbered 0 to 39.

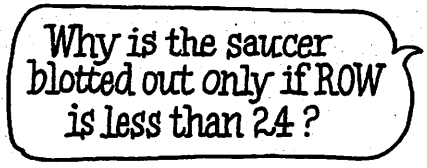
The next program has a flying saucer coming down the screen and hovering just above the ground. Change to MODE 4 or MODE 6 before you type it in, because if you are in MODE 7 some of the characters in line 20 will not look right on the screen.

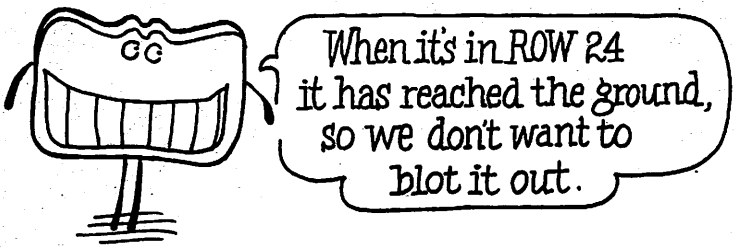
```
10 MODE 4
20 LET F$ = "_/*\_"
30 PRINT TAB(0,25) "_____
                    "
_____"
40 FOR ROW = 0 TO 24
50 PRINT TAB(18,ROW) F$
60 FOR W = 1 TO 500
70 NEXT W
80 IF ROW < 24 THEN PRINT TAB(18,ROW)
"      "
90 NEXT ROW
```



The flying saucer, which is made up of characters you will find on the keyboard, is stored in FS$. The horizontal line _ is the one on the same key as the pound sign. You will need it again in line 30, for the surface of the earth.

As in the truck program, we have a FOR . . . NEXT loop to move the saucer. This time it is the row that changes, not the column. There are five spaces between the quote marks in line 80. These are used to blot out one flying saucer before the next one is printed.





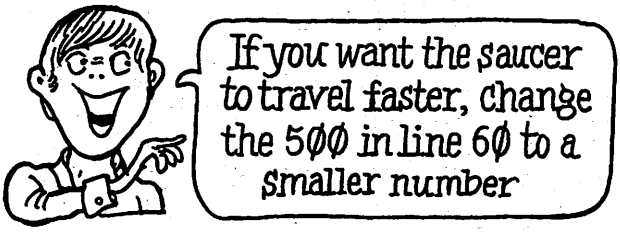
To make the saucer take off again, change line 40, so that it reads

```
40 FOR ROW = 24 TO 0 STEP -1
```

and make line 80:

```
80 PRINT TAB(18,ROW) "      "
```

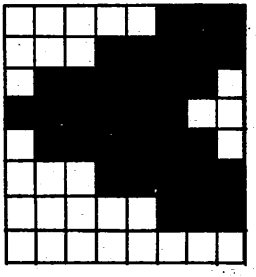
STEP −1 tells the computer to count backwards, subtracting one each time, so the saucer starts at row 24 and finishes at row 0.



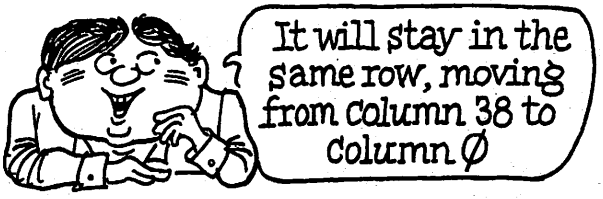
In a later chapter we are going to add some sound effects to this program. If you record it on to a cassette now, you will not have to type it in again. (See Chapter 32)

For you to do

Put the computer into MODE 4. Then instruct it to give this rocket the code number 235:



Now make the rocket fly across the screen, from right to left.



CHAPTER 27

Colours

If you are using a colour T.V. set, you would, no doubt, like to see your cars, planes and invaders in colour. How many different colours you can have on the screen at a time depends on which MODE you are in.

Let's first try out MODE 5, in which you can use four colours. Using the word COLOUR you can tell the computer what colour you wish the background to be – the sky behind your aeroplane, for example.

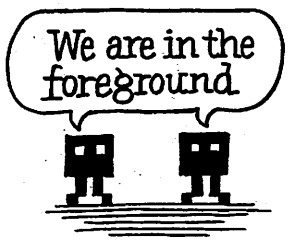
Try this:

```
MODE 5      [RETURN]
COLOUR 129      [RETURN]
CLS     [RETURN]
```

Your commands to the computer for background colours are:

COLOUR 128	black background
COLOUR 129	red background
COLOUR 130	yellow background
COLOUR 131	white background

Now for the foreground, which means any characters that are printed on the screen.



Type

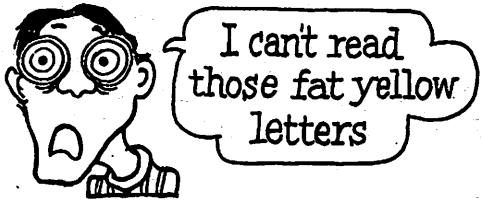
```
COLOUR 2        [RETURN]
```

The prompt and the cursor are now yellow, as will be any characters you now type in.

The instructions for foreground colours are:

`COLOUR 0`	black foreground
`COLOUR 1`	red foreground
`COLOUR 2`	yellow foreground
`COLOUR 3`	white foreground

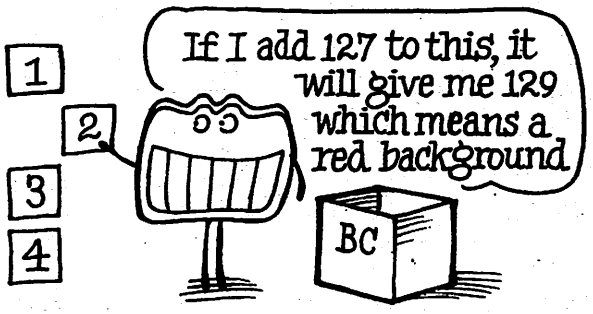
Here is a program which will show you what the different background and foreground colours look like together. Before you start typing it in, press BREAK and you will be back in MODE 7, with black screen and white letters.



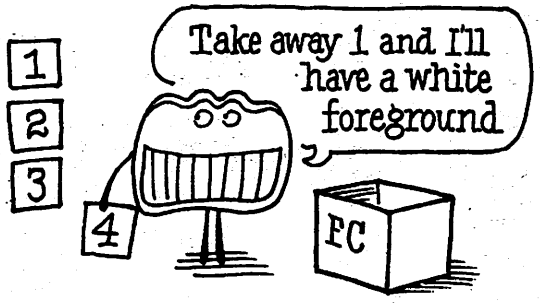

```
10 MODE 5
20 FOR T = 1 TO 120
30 LET BC = RND(4) + 127
40 LET FC = RND(4) - 1
```

```
50 COLOUR BC
60 COLOUR FC
70 PRINT "MICRO";
80 NEXT T
```

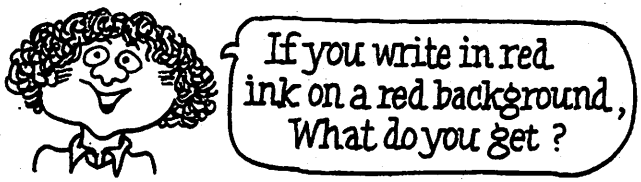
At line 30 the computer chooses a number for the background colour. We need a number between 128 and 131, so it chooses one between 1 and 4 and then adds 127 to it.



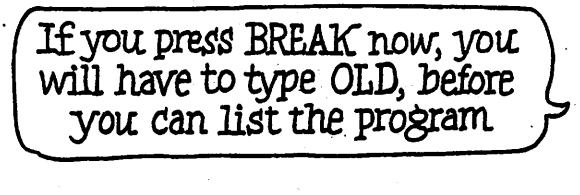
At line 40 it chooses the foreground colour. This time we need a number between 0 and 3.



Each time the computer goes round the FOR . . . NEXT loop it chooses the two colours at random and prints "MICRO". Why are there some places where it doesn't seem to have printed anything?

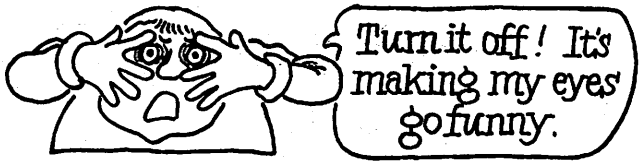


Change to MODE 7 and tell the computer to LIST the program.



If you change three lines of the program, you can see the sixteen colours of MODE 2. There are eight still and eight flashing.

```
10 MODE 2
30 LET BC = RND(16) + 127
40 LET FC = RND(16) - 1
```



You will find details of the various colours the different MODEs give you on page 170.

Here is the truck program from Chapter 26 again, but this time in colour. The lines which have been added are 15 and 18.

```
10 MODE 5
15 COLOUR 130 : CLS
18 COLOUR 0
20 VDU 23,230,255,255,255,255,255,255
,96,96
30 VDU 23,231,224,224,239,239,239,255
,198,198
40 LET TRUCK$ = " " + CHR$(230) + CHR
$(231)
50 FOR COLUMN = 1 TO 17
60 PRINT TAB(COLUMN,15);TRUCK$
70 FOR W = 1 TO 500
80 NEXT W
90 NEXT COLUMN
```

Line 15 has two instructions in it, COLOUR 130 and CLS. You can put more than one instruction in a line if you separate them with colons.

For you to do

Write a program in which you tell the computer to:

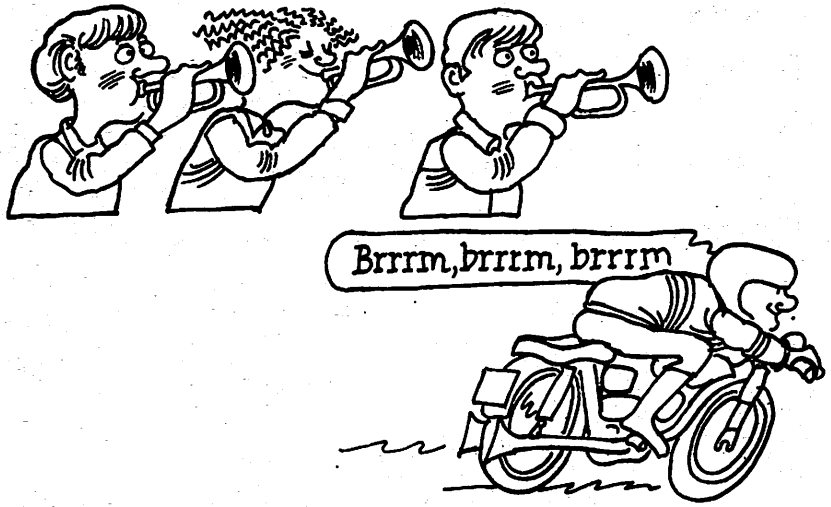
Go into MODE 5.
Choose a white background.
Clear the screen.
Choose red for the foreground colour.
Print your name on the left of the screen and make it move slowly over to the right.

Two FOR...NEXT loops needed here : one inside the other

CHAPTER 28

Sounds

Your micro has four 'channels', through which it can send out sounds. They are numbered from 0 to 3. Channels 1, 2 and 3 play musical notes, whereas channel 0 makes 'noises', such as you might use for sound effects in games.



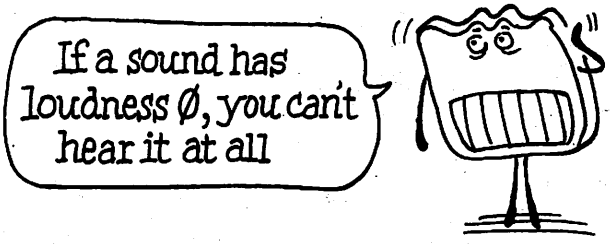
Let's start with a musical note. The BASIC keyword for 'make a sound' is SOUND. We have to tell the computer four things about the sound we want to make:

Which channel it is to be played on,
How loud it is to be,
How high or low it is to be (the 'pitch'),
How long it is to play for.

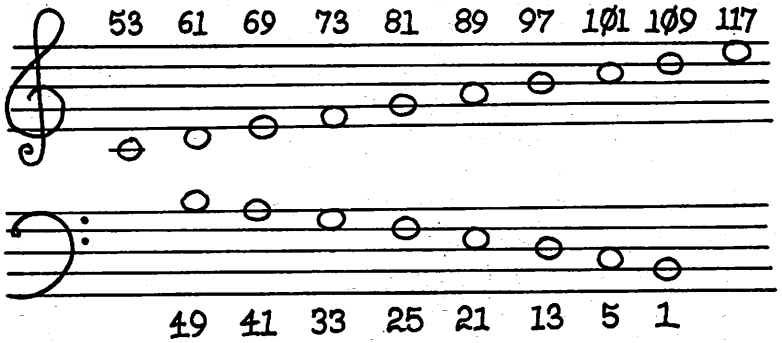
Try this note:

```
SOUND 1,-15,53,20        [RETURN]
```

The loudness numbers go from −15, which is loudest sound the BBC Micro can play, to 0, which is the quietest.



The lowest note the micro can play is pitch 0, and the highest is 253. Pitch 53 is Middle C. For those of you who can read music, this diagram will help you to write your own tunes:



If you need a sharp or a flat, this is how to work out its pitch number: As B flat is halfway between A and B, its pitch number is halfway between 89 and 97, which is 93.

The last number in our SOUND instruction is the length of the note. 20 makes the computer play for about one second.

Now here is a music program for you to try:

```
10 FOR A = 1 TO 2
20 SOUND 1,-7,73,10
30 SOUND 1,-7,81,10
40 SOUND 1,-7,89,10
50 SOUND 1,-7,73,10
60 SOUND 1,0,0,1
70 NEXT A
80 FOR B = 1 TO 2
90 SOUND 1,-7,89,10
100 SOUND 1,-7,93,10
110 SOUND 1,-7,101,20
120 SOUND 1,0,0,1
130 NEXT B
```

Lines 60 and 120 do not play a note. They just make a short gap between each phrase of the song. Take them out, RUN the program and see what the difference is.

If we use channel 1 and channel 2 in the same program we can play chords, with two notes playing at the same time. Here is a program which plays the first two bars of a Christmas carol:

```
10 SOUND 2,-10,49,20
20 SOUND 1,-10,61,20
30 SOUND 2,-10,41,20
40 SOUND 1,-10,77,20
50 SOUND 2,-10,33,40
60 SOUND 1,-10,81,40
70 SOUND 2,-10,49,40
80 SOUND 1,-10,81,10
90 SOUND 1,-10,77,10
100 SOUND 1,-10,81,10
110 SOUND 1,-10,89,30
```

```
120 SOUND 2,-10,53,20
130 SOUND 2,-10,49,20
140 SOUND 1,-10,81,20
```

Now for the 'noises' on channel 0.

There are eight types of sound effect the computer can give you. They are numbered 0 to 7. When you are giving the micro an instruction to make a channel 0 noise, after the word SOUND, you tell it

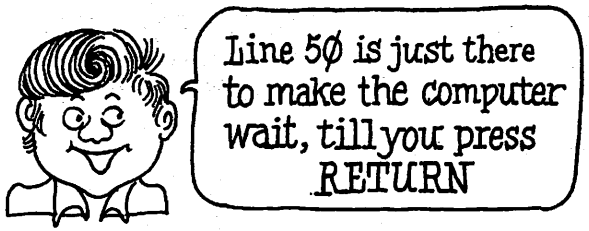
The channel number,
The loudness,
Which noise,
The length.

Try this one:

```
SOUND 0,-15,2,40
```

Here is a short program which shows you the eight types of noise. RUN the program, and after each noise press RETURN.

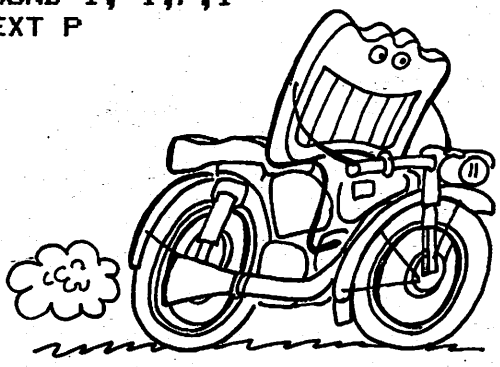
```
10 FOR EFFECT = 0 TO 7
20 CLS
30 PRINT TAB(10,15) "SOUND EFFECT ";
EFFECT
40 SOUND 0,-15,EFFECT,40
50 INPUT I$
60 NEXT EFFECT
```



You will probably think that effects 3 and 7 sound like an old machine gun, which is on its last legs. These two noises are not very useful on their own, but when one of them is playing together with channel 1, you can have some interesting effects.

Here is quite a realistic impression of a motorcycle driving off.

```
10 SOUND 0,-15,3,100
20 FOR P = 100 TO 200
30 SOUND 1,-1,P,1
40 NEXT P
```

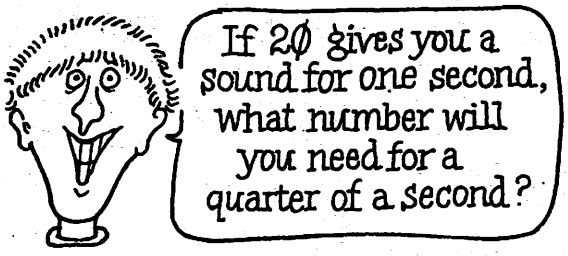


Line 10 tells the computer we want channel 0, loud, noise number 3, for five seconds.

Line 30, inside the FOR . . . NEXT loop, plays a very quiet, short note on channel 1. As the computer goes round the loop, the pitch of the note (P) gets higher and higher.

For you to do

Look back at the flying saucer program on p. 98. Put in an extra line (line 55) so that every time the saucer moves down, the computer makes a channel 0 noise, quite loud, for a quarter of a second.



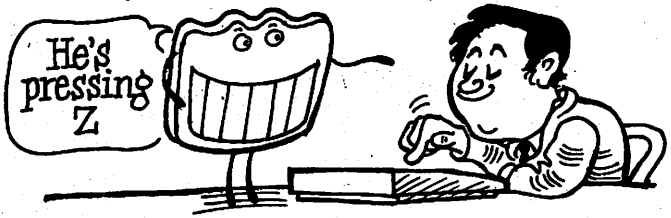
Try noises 0, 1, and 2 and see which you like best.

CHAPTER 29

GET$ and INKEY$

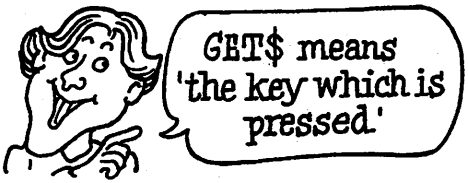
When you are playing a game on the computer, it would be very tedious if you had to keep pressing RETURN after you had snapped an alien.

The BBC Micro has two very useful BASIC keywords which tell it to look at the keyboard and see which key is being pressed. They are GET$ and INKEY$.



Here is a short program to show you how GET$ works:

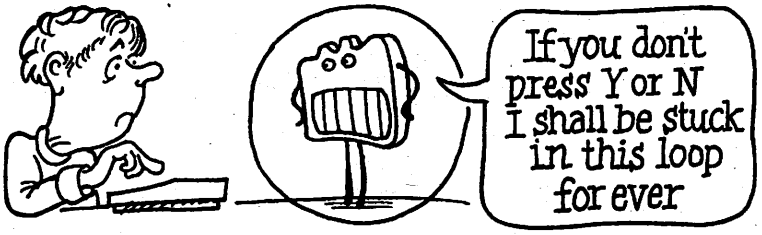
```
10 CLS
20 PRINT "PRESS KEY Z"
30 IF GET$ = "Z" THEN PRINT    "THANK
YOU" ELSE PRINT "I WANTED Z"
```



When it reaches line 30, the computer will wait until you press a key. As soon as you do, it will print its message.

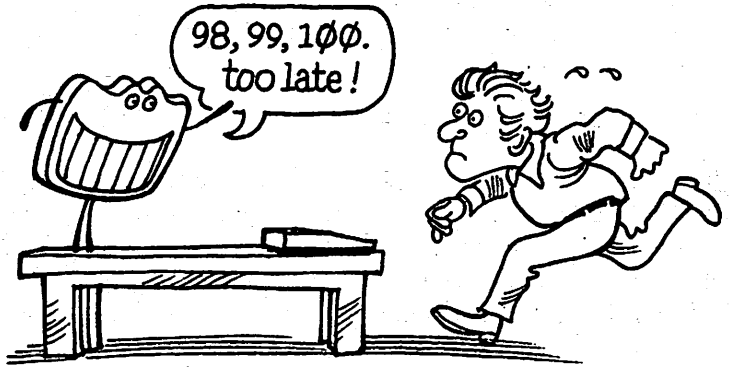
When you have finished playing a computer game, you may like to have another turn. If you put these five lines at the end of a program, the computer will ask you if you want to play again, and then wait for you to press a key. If you press Y it will RUN the program again. Can you work out what happens if you press N? What happens if you press any other key?

```
200 PRINT TAB(0,30) "PLAY AGAIN (Y OR
N)"
210 REPEAT
220 LET A$ = GET$
230 IF A$ = "Y" THEN RUN
240 UNTIL A$ = "N"
250 END
```



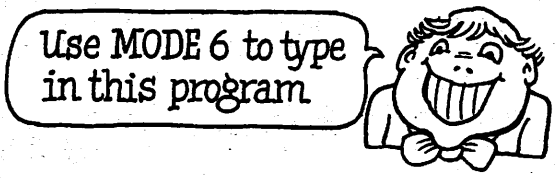
INKEY$ is slightly different. When the computer sees this word, it will not wait for ever, as it would with GET$. After the word INKEY$, you must give the computer a number, in brackets. If you put INKEY$ (100), the computer will wait and count up to 100. If you haven't pressed a key by then, it will continue with the program. Try this:

```
10 CLS
20 PRINT "PRESS Z"
30 IF INKEY$(100) = "Z" THEN PRINT
"THANK YOU" ELSE PRINT "TOO LATE"
```



Change the (100) to (50) and you will have even less time to press Z.

Here is a program called "COPTER" in which you have to land a helicopter on a small landing pad in the sea. The helicopter starts at the top of the screen. You guide it down, using key X to move right and key Z to move left.



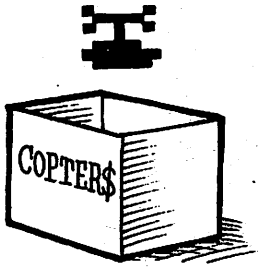

```
10 MODE 5
20 VDU 23;8202;0;0;0;
30 VDU 23,250,130,124,146,16,124,255,
126,0
40 LET COPTER$ = CHR$(250)
50 LET COPCOL = RND(18)
60 LET PADCOL = RND(18)
70 PRINT TAB(0,28) "~~~~~~~~~~~~~~~~~~~~
~~~~"
```

```
  80 PRINT TAB(PADCOL,28) "-"
  90 FOR ROW = 0 TO 28
 100 PRINT TAB(COPCOL,ROW) COPTER$
 110 LET DIRECTION$ = INKEY$(10)
 120 IF ROW < 28 THEN PRINT TAB(COPCOL,
ROW) " "
 130  IF DIRECTION$ = "X" AND COPCOL <
 19 THEN LET COPCOL = COPCOL + 1
 140 IF DIRECTION$ = "Z" AND COPCOL > 0
 THEN LET COPCOL = COPCOL - 1
 150 NEXT ROW
```

Let's look at the program a part at a time to see what each line does:

```
  10 MODE 5
  20 VDU 23;8202;0;0;0;
  30 VDU 23,250,130,124,146,16,124,255,
126,0
  40 LET COPTER$ = CHR$(250)
```

Lines 30 and 40 design the helicopter, giving it code number 250, and put it into the string variable box COPTER$.



Line 20 is a special command to the BBC Micro, telling it to turn off the flashing cursor. We don't want it on the screen in a program like this. After you have RUN the program you can turn the cursor back on by changing MODE.

```
50 LET COPCOL = RND(18)
60 LET PADCOL = RND(18)
```

Here the computer chooses the column the helicopter is to start in (COPCOL), and the column the landing pad is to be in (PADCOL).

```
   70 PRINT TAB(0,28) "~~~~~~~~~~~~~~~~~~~~
~~~~"
   80 PRINT TAB(PADCOL,28) "-"
```

Line 70 prints the sea at the bottom of the screen, and line 80 puts in the landing pad.

```
90 FOR ROW = 0 TO 28
```

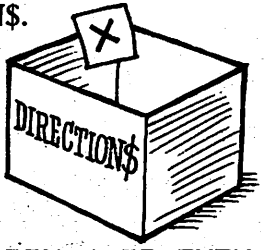
The helicopter will move down the screen, from row 0 to row 28.

```
100 PRINT TAB(COPCOL,ROW) COPTER$
```

The computer prints the helicopter.

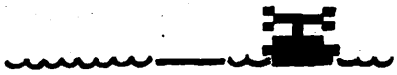
```
110 LET DIRECTION$ = INKEY$(10)
```

Here the computer counts to ten. If you press a key before it reaches ten, the character on that key goes into DIRECTION$.



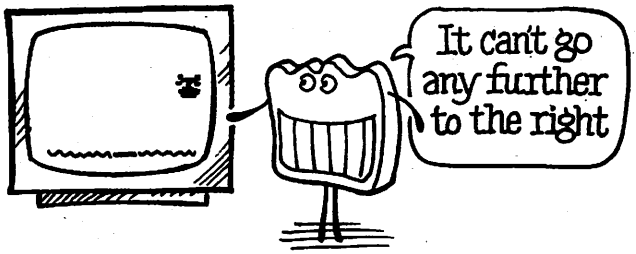

```
  120 IF ROW < 28 THEN PRINT TAB(COPCOL,
ROW) " "
```

This line blots out the helicopter by printing a space over it. If it has reached row 28 it is not blotted out, because it has landed.



```
130  IF DIRECTION$ = "X" AND COPCOL <
19 THEN LET COPCOL = COPCOL + 1
```

Here the computer looks into the box DIRECTION$ to see whether you have pressed X. It also checks to see whether the helicopter is in column 19, which is the one on the very right of the screen. If you *have* pressed X, and if the helicopter *is* in a column with a lower number than 19, the computer arranges for the next helicopter to be one column further to the right.




```
140 IF DIRECTION$ = "Z" AND COPCOL > 0
THEN LET COPCOL = COPCOL - 1
```

This line arranges for the helicopter to move left, as long as that won't mean it goes off the screen.

```
150 NEXT ROW
```

Here is the end of the 'FOR . . . NEXT' loop, which began at line 90.

For you to do

Put in some COLOUR instructions for the computer, telling it what colour to make the sky, the helicopter, the sea and the landing pad.

Now add some SOUND. Perhaps the helicopter could make a noise as it is coming down to land.

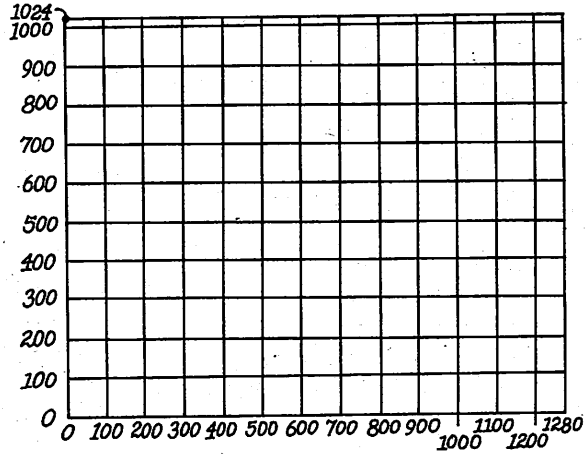
Add some lines at the end of the program, in which the computer asks you if you would like to play again. If you would, it RUNs the program again.

CHAPTER 30

Draw and paint

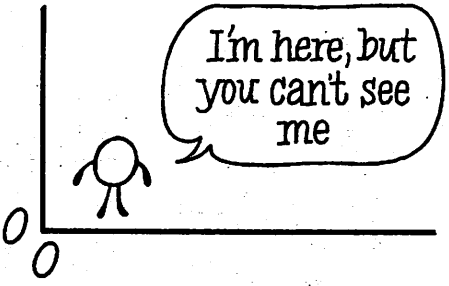
In MODES 0, 1, 2, 4 and 5, we can make pictures by drawing lines on the screen. When we do this we are using 'high resolution graphics', which means drawing accurate pictures or diagrams.

Again you need to think of the screen as being ruled into squares. Here is a 'hi-res' screen, with the co-ordinate numbers written in:



Notice that the 'across' co-ordinates start from 0 at the left, and the 'up' co-ordinates begin with 0 at the bottom.

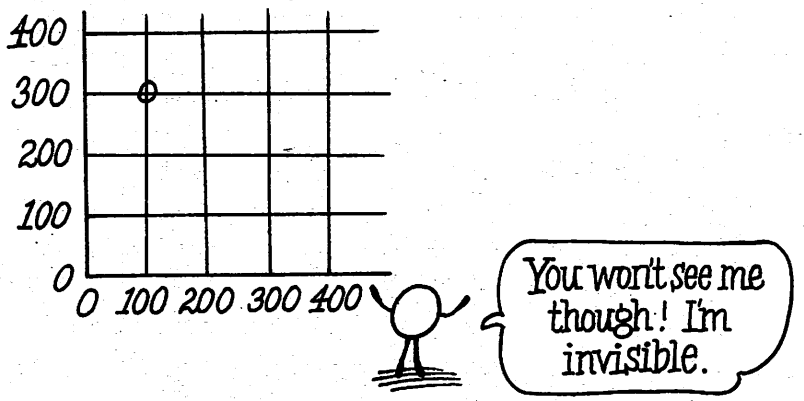
On the screen is an invisible 'graphics cursor'. When you switch on your micro, this cursor is down in the bottom left-hand corner of the screen, at 0,0.



To move the graphics cursor we use the BASIC word MOVE, and to tell it to draw a line we use DRAW. To tell the graphics cursor which point on the screen you want it to move to, you give it the co-ordinates – the 'across' number, then a comma, then the 'up' number.

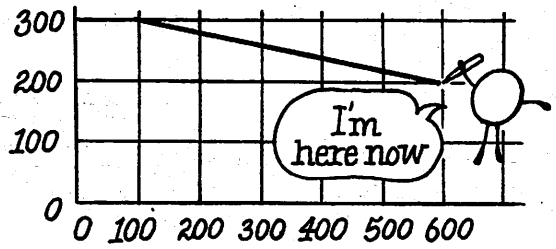
```
MOVE 100,300        [RETURN]
```

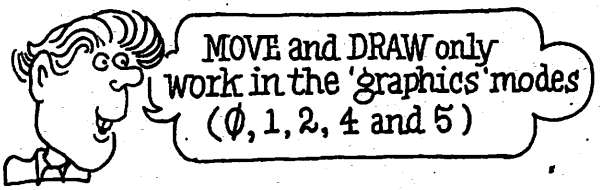
tells the cursor to move to here:



To draw a line from here to 600, 200, the instruction is

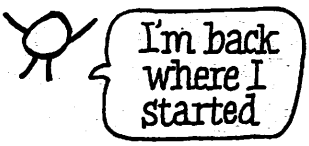
```
DRAW 600,200        [RETURN]
```





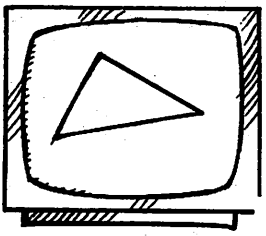
Here is a program which will draw a square in the middle of the screen:

```
10  MODE 4
20  MOVE 500,300
30  DRAW 900,300
40  DRAW 900,700
50  DRAW 500,700
60  DRAW 500,300
```



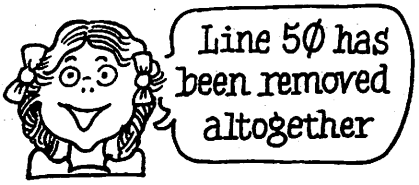
Now for a triangle. Don't forget to clear the memory before you type in the program.

```
10 MODE 4
20 MOVE 400,300
30 DRAW 900,400
40 DRAW 500,600
50 DRAW 400,300
```



If you want to paint in your triangle, so that it looks solid, instead of just an outline, you can use the instruction PLOT 85. Change the triangle program so that it reads

```
10 MODE 4
20 MOVE 400,300
30 DRAW 900,400
40 PLOT 85,500,600
```



A triangle is made by joining three points together. If we tell the graphics cursor

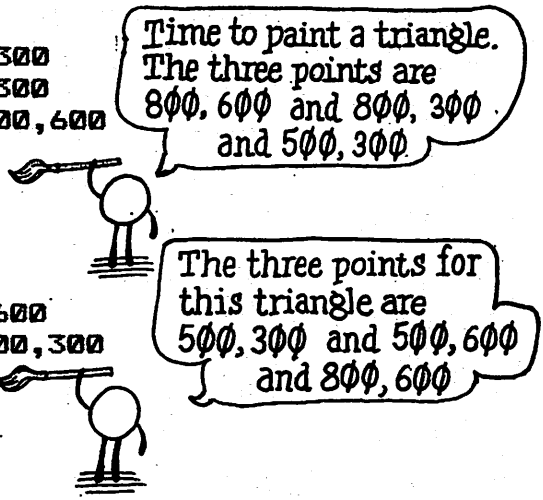
```
PLOT 85,500,600
```

it draws a line to 500, 600. It then looks back at the last two points it visited. Using them as the other two corners it paints in the triangle.

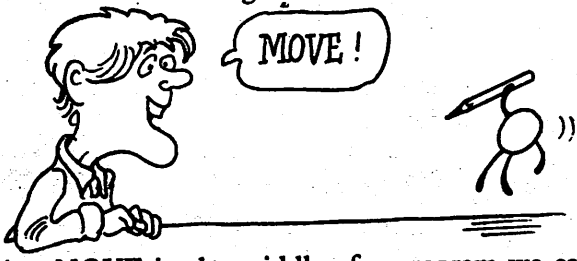
You can make a solid square by drawing two solid triangles.

Here is a program for it:

```
10 MODE 4
20 MOVE 500,300
30 DRAW 800,300
40 PLOT 85,800,600
50 DRAW 500,600
60 PLOT 85,500,300
```


Time to paint a triangle. The three points are 800, 600 and 800, 300 and 500, 300
The three points for this triangle are 500, 300 and 500, 600 and 800, 600

When the graphics cursor is told to MOVE, it moves to the new point without drawing a line behind it.



By using MOVE in the middle of a program we can draw a shape inside another shape. See what this program does:

```
10 MODE 4
20 MOVE 200,200
30 DRAW 700,200
```

```
40 DRAW 700,700
50 DRAW 200,700
60 DRAW 200,200
70 MOVE 300,300
80 DRAW 600,300
90 PLOT 85,450,600
```

Now for some colour. The BASIC colour word to use with high resolution graphics is GCOL0

In MODE 5, the instructions for coloured lines and painting are these:

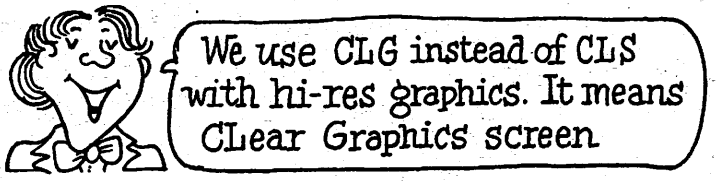
`GCOL0,0`	black
`GCOL0,1`	red
`GCOL0,2`	yellow
`GCOL0,3`	white

You can change the colour of the background behind your drawings too, with

```
GCOL0,128      black background
GCOL0,129      red background
GCOL0,130      yellow background
GCOL0,131      white background
```

Let's put colour into the program on p. 123. Change line 10 to

```
10 MODE 5: GCOL0,130: CLG
```



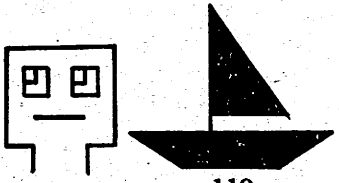
Put in

```
15 GCOL0,0
65 GCOL0,1
```

Now RUN the program again.

For you to do

Write programs to draw these pictures. Draw them on squared paper first, putting the 'across' and 'up' co-ordinates along the edges of the paper, as on p. 119. For the hull of the yacht you will need three triangles.

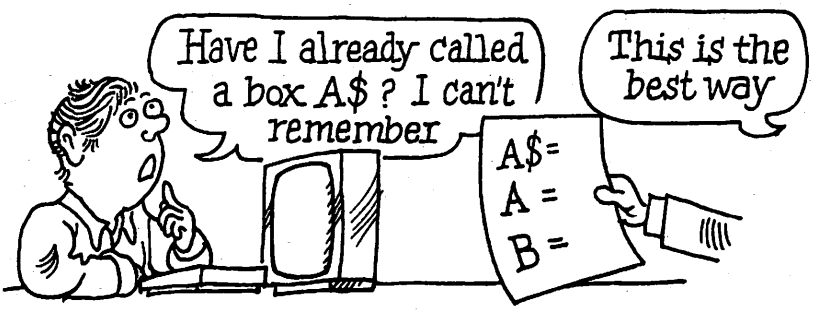


Put in some colours.

CHAPTER 31

Planning a program

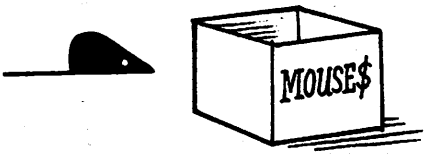
When you write a program it is best to plan it first. If you just sit down at the computer and start typing you will often find you get in a muddle, or leave out something important.



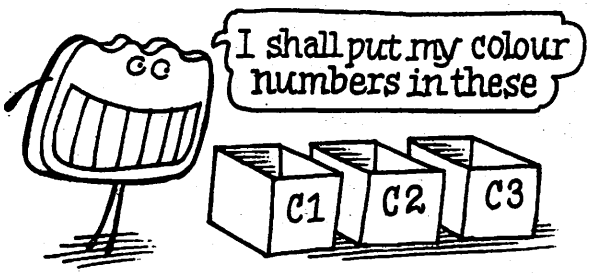
We are going to plan a program called "Three Mice". First of all we'll write down what we want it to do.

We'll need to design a mouse character first. Then the computer will choose three colours. It will show three mice on the screen, one at a time, one of each colour. Then it will ask you what colours they were. You enter the numbers for the three colours. The computer checks to see if you were right. If you were, it covers the screen with "WELL DONE", and plays two musical notes. If you were wrong, it prints "WRONG" in the middle of the screen, and plays a long, low note.

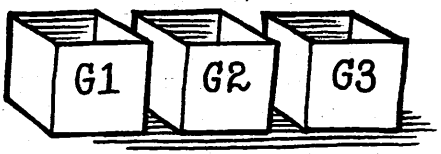
Now we will make a list of the variables, or boxes we will use. The mouse will go into MOUSE$.



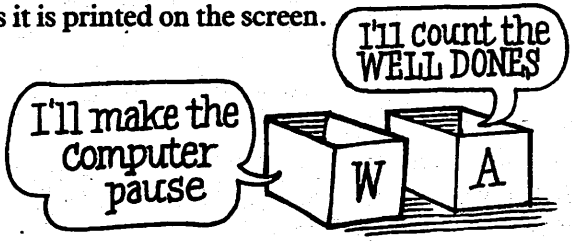
The three colours the computer chooses will go into C1, C2 and C3.



Your guesses at the three colours will go into G1, G2 and G3.



We shall need a FOR . . . NEXT loop to make the computer pause while it shows each mouse. Otherwise the mice will flash on and off the screen so quickly you won't see them. We shall need another FOR . . . NEXT loop to count "WELL DONE" as it is printed on the screen.



Here is the complete list:

String variable

MOUSE$ a mouse

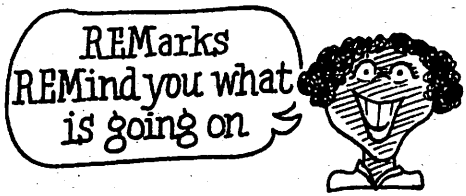
Number variables

C1 the colour of the first mouse
C2 the colour of the second mouse
C3 the colour of the third mouse
G1 your guess at the first colour
G2 your guess at the second colour
G3 your guess at the third colour

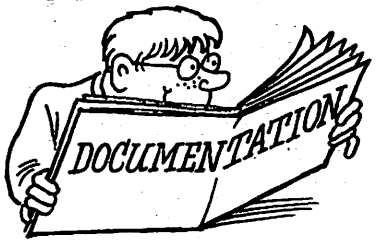
Loop variables

W loop for making the computer pause when showing a mouse
A loop for counting "WELL DONE"s

When you are typing in the program, you can put in extra lines, called 'remarks', to remind yourself what each part of the program does. The computer won't take any notice of these lines, but you will find them useful when you look back at your program later on. The BASIC word for 'remark' is REM.

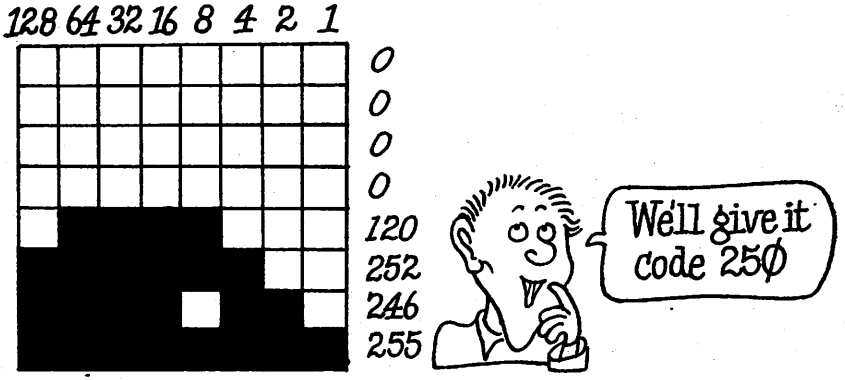


When you have decided what the program is to do, and thought about the variables you will need, write down your program lines on paper before you type them into the

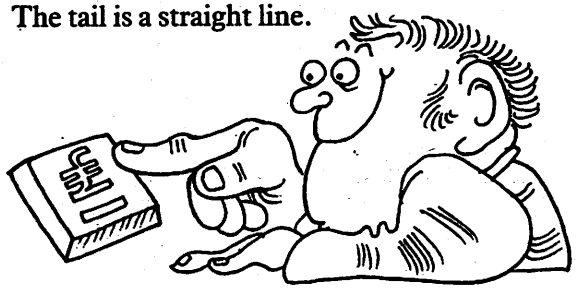


computer. This will save you time in the long run. The description of the program, together with the list of variables and the program listing, is called the 'documentation'. Keep it in a safe place, as it will be useful to look back at if you should later want to change anything in the program.

Now for the "Three Mice" program. Here is the body of the mouse:



The tail is a straight line.



The first section of the program reads:

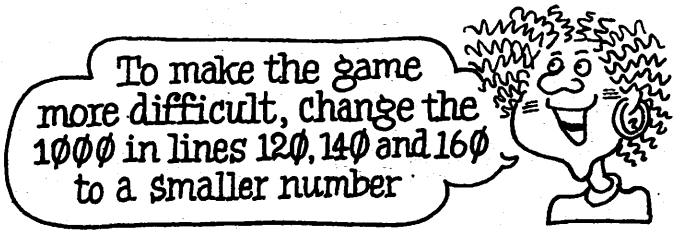
```
10 REM THREE MICE
20 REM THE MOUSE
30 VDU 23,250,0,0,0,0,120,252,246,255
40 LET MOUSE$ = "_" + CHR$(250)
```

Now the computer chooses the colours.

```
   50 REM THE COMPUTER CHOOSES THREE
COLOURS
   60 LET C1 = RND(3)
   70 LET C2 = RND(3)
   80 LET C3 = RND(3)
```

And shows the mice:

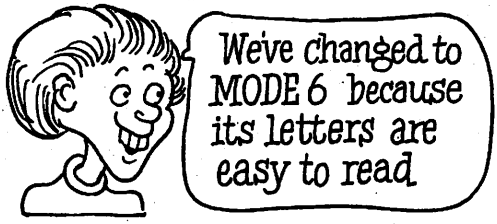
```
 90 REM THE COMPUTER SHOWS THE MICE
100 MODE 5
110 COLOUR C1: PRINT TAB(9,15) MOUSE$
120 FOR W = 1 TO 1000: NEXT W: CLS
130 COLOUR C2: PRINT TAB(9,15) MOUSE$
140 FOR W = 1 TO 1000: NEXT W: CLS
150 COLOUR C3: PRINT TAB(9,15) MOUSE$
160 FOR W = 1 TO 1000: NEXT W: CLS
```



Now you have to guess the colours:

```
170 REM YOU GUESS
180 MODE 6
190 PRINT "FOR RED, ENTER 1"
200 PRINT "FOR YELLOW, ENTER 2"
210 PRINT "FOR WHITE, ENTER 3"
220 PRINT TAB(0,10) "FIRST COLOUR?"
```

```
230 INPUT G1
240 PRINT "SECOND COLOUR?"
250 INPUT G2
260 PRINT "THIRD COLOUR?"
270 INPUT G3
280 MODE 5
```



Next the computer has to check your numbers to see whether they were all correct. If they were it jumps to line 350.

```
290 REM THE COMPUTER CHECKS YOUR
NUMBERS
300 IF G1 = C1 AND G2 = C2 AND G3 = C3
THEN GOTO 350
```

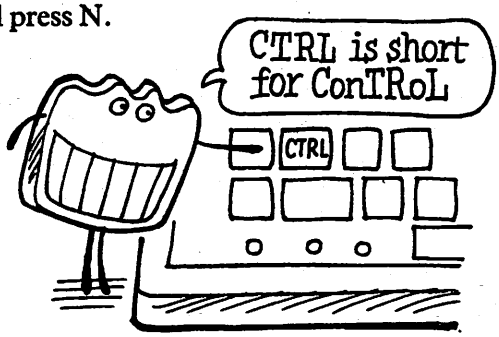
The next four lines tell the computer what to do if you were wrong.

```
310 REM WRONG
320 PRINT TAB(7,16) "WRONG"
330 SOUND 1,-15,5,40
340 END
```

This is what happens if you were right:

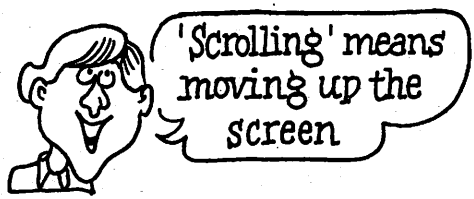
```
350 REM RIGHT
360 COLOUR 2
370 FOR A = 1 TO 60
380 PRINT "WELL DONE ";
390 NEXT A
400 SOUND 1,-15,53,10
410 SOUND 1,-15,101,10
```

When you tell the computer to LIST a long program like this, you will see it whiz up the screen so fast you can't read the first part. To see a 'page' at a time, hold down the CTRL key and press N.



You won't see anything happening on the screen when you press CTRL N, but now when you tell the computer to LIST, it will show you just the first screenful, or 'page', of your program. Press SHIFT if you want to see the next page, or ESCAPE if you don't.

If you want to leave 'paged mode', and go back to normal scrolling, press CTRL and O (the letter O, not the number 0).



For you to do

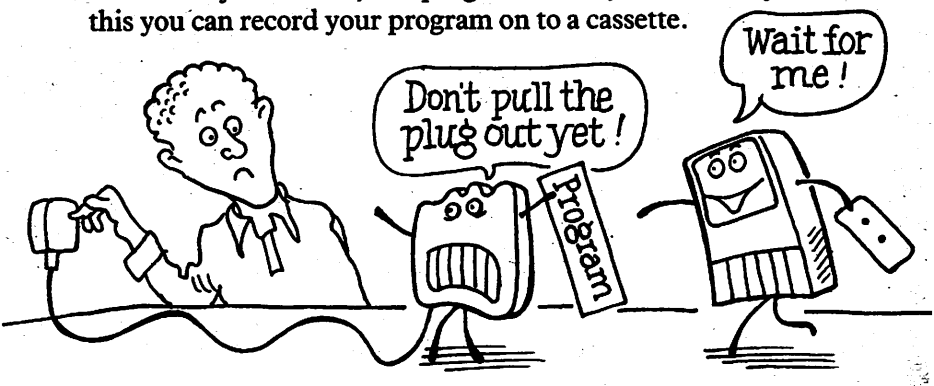
Change the 'RIGHT' part of the program (lines 350 to 410) so that instead of printing 'WELL DONE', the computer prints a screenful of mice. Perhaps you can arrange for the computer to choose a random colour for each mouse.

Now change the 'WRONG' part (lines 310 to 340) so that we see a mouse running across the screen.

CHAPTER 32

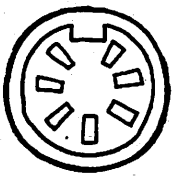
Saving your program on cassette

When you have typed in quite a long program, you will probably want to keep it, to RUN again later. When you switch off your micro, the program is lost, but before you do this you can record your program on to a cassette.

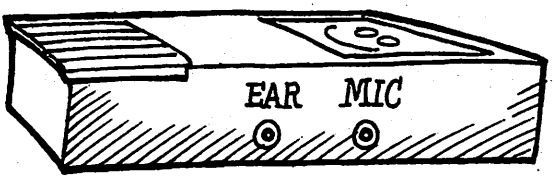


You will need a cassette recorder, a blank cassette, and a lead to go from your micro to the cassette recorder. A portable mono cassette recorder should work well, with a C10, C12 or C15 cassette.

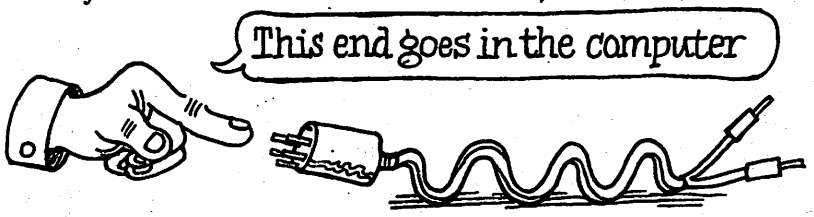
Your BBC Micro has a socket at the back marked 'cassette'. It is called a seven-pin DIN socket and looks like this:



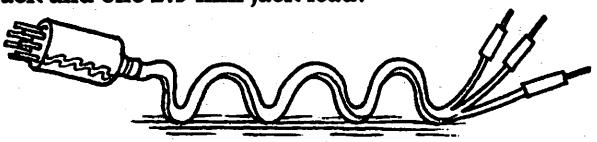
On your cassette recorder you will probably have two small sockets, called 'jack sockets', like this:



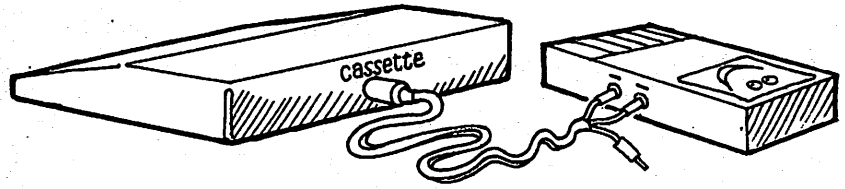
If so you will need a 5 DIN to two 3.5 mm jack lead.



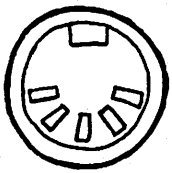
If your cassette recorder also has a smaller jack socket marked REM or REMOTE, you can use a 7 DIN to two 3.5 mm jack and one 2.5 mm jack lead.



Plug in the two larger jack plugs to your EAR and MIC sockets. Unfortunately, the manufacturers do not label them to tell you which goes in EAR and which in MIC. Often, the plug for the EAR socket has a black lead, but not always. However, it wouldn't damage the computer or the cassette recorder if you put them in the wrong way round.



If you don't have jack sockets on your cassette recorder you will have a 5 pin DIN socket.



and will need a 5 DIN to 5 DIN lead, or 7 DIN to 5 DIN and one 2.5 mm jack.

Don't worry about plugging in the small 'remote' jack for the moment.

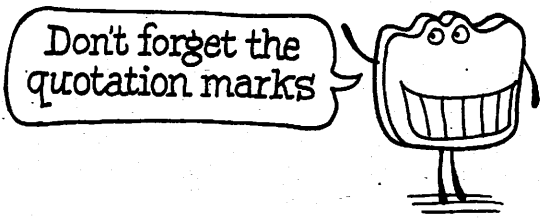
Plug your cassette recorder into the electricity (unless it runs on batteries). Put in the cassette. Wind it back to the beginning and then move it forward just a little so that the brown tape is showing, not the coloured plastic leader tape. Set the volume control to fairly loud (about 7, if it has 10numbers). If there is a tone control, put it to high.

LIST your program on the screen. Make sure it is just as you want it. Now you must give it a "file-name". You can use letters and numbers for your file-names. It must not be longer than ten characters.

Suppose you have in the memory the flying saucer program from Chapter 25, and want to save it on cassette. We can't call it "FLYING SAUCER" as that is more than ten characters, and spaces aren't allowed. Let's call it "SAUCER".

Type in

```
SAVE "SAUCER"        [RETURN]
```



Now you see the message

```
RECORD then RETURN
```

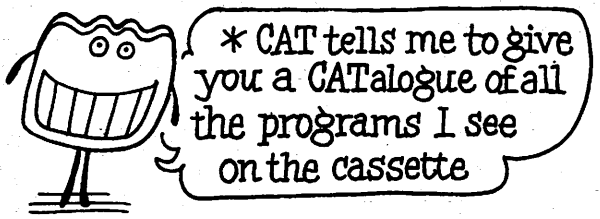
Set the cassette recorder to record, and then press RETURN. After a few seconds, the name of your program will come up on the screen, followed by numbers and letters. When you hear a 'beep' sound, and you see the prompt back on the screen, stop the cassette recorder.

```
SAUCER      00 00BD
>
```

Now we'll check that the program has actually been recorded. Rewind the cassette to the beginning. Type in

```
*CAT        [RETURN]
```

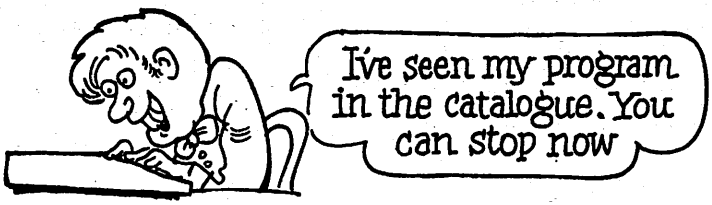
and press PLAY on the cassette recorder.



Soon you should see your program's file name, followed by numbers and letters. When the computer has checked the whole program there will be six digits on the screen, and you can stop the cassette recorder.

```
SAUCER      00 00BD
```

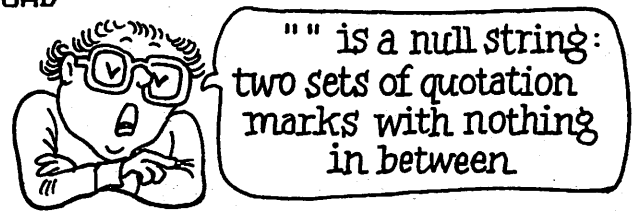
The computer will not stop looking at the cassette after it has passed your program. It will go on scanning the cassette for any more programs. To make it stop, press ESCAPE.



If your program's name did not appear on the screen, something must have gone wrong. You may have put the two jack plugs in the wrong sockets of the cassette recorder, or you may need to turn the volume up a little. If your program did not SAVE successfully, turn back to p. 135 and try again.

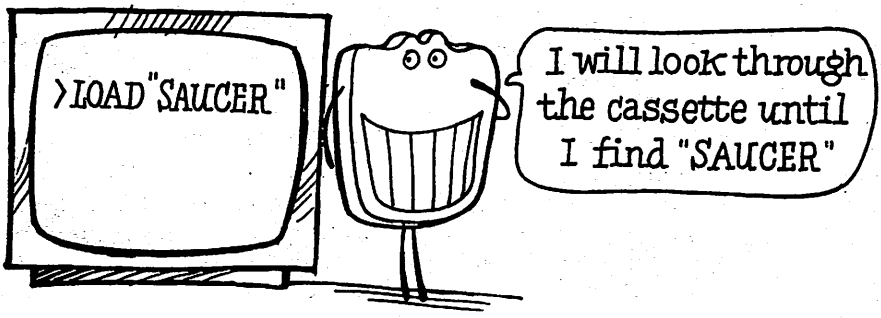
With your program safely on cassette, you can turn the computer off, or write another program.

When you want to put your program back into the computer you can use the BASIC word LOAD. One way is to type `LOAD""`



which will LOAD the first program the micro finds on the cassette.

The other way is to type LOAD and then the name of your program, with quotation marks round it.



If you have more than one program on a cassette, the second way is better, as it will find the right program.

Rewind the cassette, press RETURN, and set the recorder to play. You will see the message

```
Searching
```

and then, when the computer has found the program,

```
Loading
```

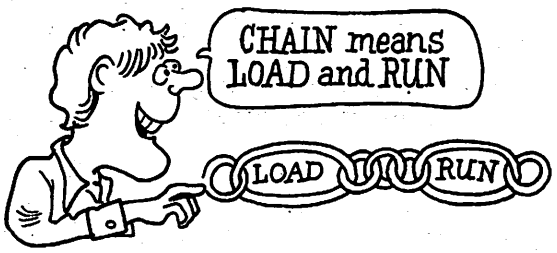
Next comes the program's name, and its six numbers or letters. As soon as the prompt appears underneath, and you hear a 'beep', you can stop the cassette recorder and LIST or RUN the program.

Instead of

```
LOAD""      or  LOAD "SAUCER"
```

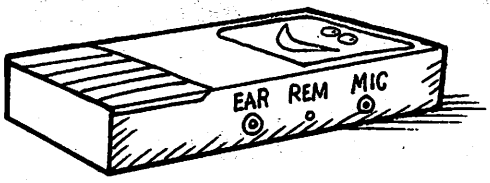
you can tell the computer to

```
CHAIN""     or  CHAIN "SAUCER"
```



This will make the program RUN automatically as soon as it is LOADed in.

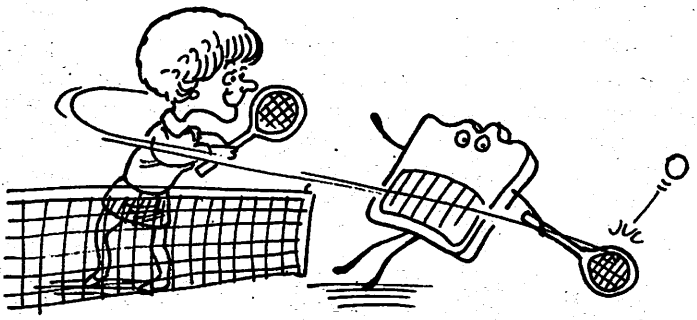
When your programs are saving and loading successfully, you can try using the small jack plug in your cassette recorder's REMOTE socket (if it has one).



With this plugged in, the computer will make the cassette recorder stop automatically when a program has been saved or loaded. The cassette recorder will only work when the BBC Micro's 'cassette motor' light is on. You may finnd that you want to rewind the cassette ready to LOAD a program, but can't, because the red light is off. If this happens, tell the computer to

```
LOAD""
```

Press RETURN and set the cassette player to 'rewind'. When it is back to the beginning of the tape, set it to 'play'.



CHAPTER 33

Games

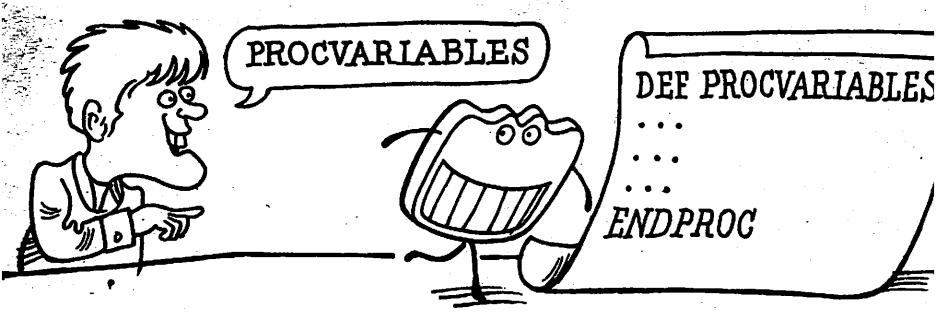
JACKPOT!

Some games test your powers of observation. *Three Mice* from Chapter 31 is one of these. In others, like *Copter*, from Chapter 29, you need quick reactions. For the first game in this chapter you need both. It is called *Jackpot!*, and it is based on the 'fruit machine' idea.

Instead of fruits, you see different coloured stars flashing in front of your eyes. When you wish to 'hold' the stars, you press H. If you then have three stars of the same colour, you score 10 points. You may 'hold' the stars 25 times, and your aim is to score 100 points.

When you type in *Jackpot!*, you will notice that it is printed in sections. This is to make it easier for you to see what each part of the program does. The first section is the main one. It gives you the name of the program, tells the computer to use MODE 5, and then turns off the flashing cursor. The next line

```
40 PROCVARIABLES
```

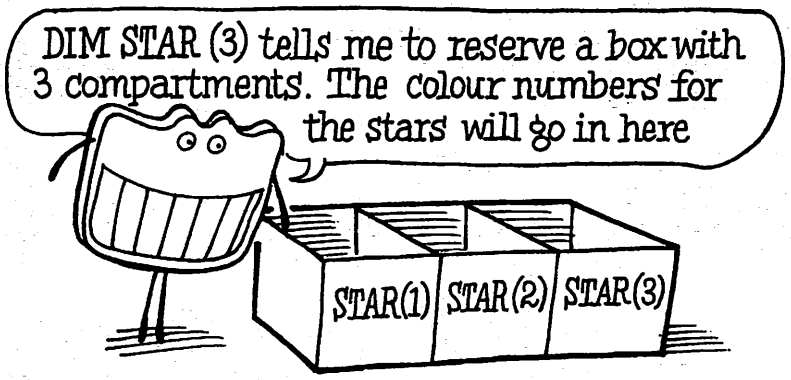


tells the computer to look for a section of the program which begins DEF PROCVARIABLES. It must go to that section and carry out the instructions it finds. When it reaches the word ENDPROC, it must come back to the main part of the program again.

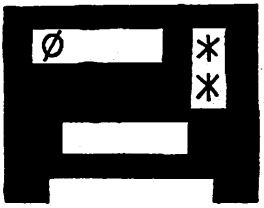
The next line, line 50, sends it off again, this time to find a section called DEF PROCSCREEN. In this way the computer carries out the various sections of the program, until it comes to the word END.

When you are typing spaces between quotation marks, make sure you type the right number. In line 170 there are 7 spaces between the quotes. In line 230 there are 3, and in lines 550 and 620 there are 15.

```
  10 REM JACKPOT!
  20 MODE 5
  30 VDU 23;8202;0;0;0;
  40 PROCVARIABLES
  50 PROCSCREEN
  60 PROCROLL
  70 PROCPLAY
  80 IF SCORE > 99 THEN PROCVICTORY
ELSE PROCTIMEUP
  90 END
 100 DEF PROCVARIABLES
 110 LET SCORE = 0: LET HOLDS = 0
 120 DIM STAR(3)
 130 ENDPROC
```



```
140 DEF PROCSCREEN
150 COLOUR 131: COLOUR 3
160 FOR ROW = 5 TO 11
170 PRINT TAB(5,ROW);"          "
180 NEXT ROW
190 FOR ROW = 12 TO 13
200 PRINT TAB(5,ROW);" ";TAB(11,ROW);
" "
210 NEXT ROW
220 COLOUR 128
230 PRINT TAB(6,6);"    ";TAB(7,9);
"    "
240 PRINT TAB(10,6);"*";TAB(10,7)"*"
250 ENDPROC
```

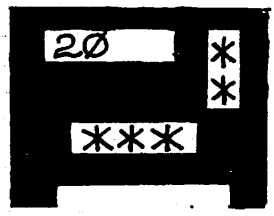


Press R to roll

```
260 DEF PROCROLL
270 COLOUR 3
280 PRINT TAB(0,17);"Press R to roll"
290 PRINT TAB(6,6);SCORE
300 REPEAT
310 UNTIL GET$ = "R"
320 PRINT TAB(0,17);"Press H to hold"
330 ENDPROC

340 DEF PROCPLAY
350 FOR S = 1 TO 3
360 LET STAR(S) = RND(3)
```

```
  370 COLOUR STAR(S)
  380 PRINT TAB(6+S,9);"*"
  390 IF INKEY$(40) = "H" THEN PROCHOLD:
PROCROLL
  400 IF SCORE > 99 OR HOLDS > 25 THEN
ENDPROC
  410 NEXT S
  420 GOTO 350
```



Press H to hold

```
  430 DEF PROCHOLD
  440 LET HOLDS = HOLDS + 1
  450 IF STAR(1) = STAR(2) AND STAR(2) =
STAR(3) THEN PROCSUCCESS
  460 ENDPROC

  470 DEF PROCSUCCESS
  480 LET SCORE = SCORE + 10
  490 SOUND 1,-10,53,20
  500 SOUND 1,-10,101,20
  510 ENDPROC

  520 DEF PROCTIMEUP
  530 COLOUR 3
  540 PRINT TAB(0,17);"                    "
  550 PRINT TAB(0,2);"Time's up!"
  560 FOR W = 1 TO 100: NEXT W
  570 SOUND 1,-10,5,50
  580 ENDPROC
```

```
590 DEF PROCVICTORY
600 COLOUR 3
610 PRINT TAB(0,17);"                    "
620 PRINT TAB(0,2);"Victory!"
630 FOR W = 1 TO 100: NEXT W
640 FOR V = 1 TO 5
650 SOUND 1,-10,101,5
660 SOUND 1,-10,105,5
670 NEXT V
680 ENDPROC
```

Variables

The two most important variables in the program are numbers:

SCORE holds the number of points you have.
HOLDS keeps a record of the number of times you have pressed H to hold the stars.

Changes you can make

Line 400 controls the length of time you have to press H, and hold the stars.

```
390 IF INKEY$(40) = "H" ...
```

To make the game easier, put a larger number than 40 after INKEY$. To make it more difficult, put a smaller number.

Other things you might like to change are the number of times you can hold the stars, and the score which gives you victory. The lines to change for these are lines 80 and 400.

Perhaps you might also like to have displayed on the screen the number of 'holds' you have had. Put in a line 295 for this.

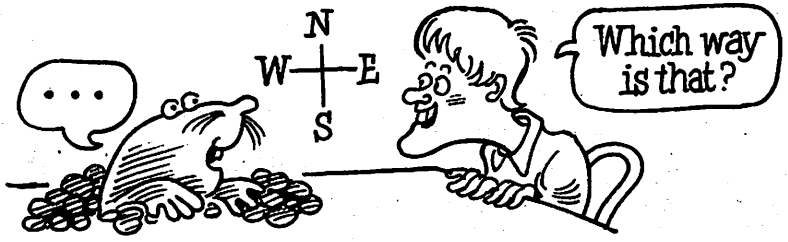
DOTTY DIGGER

A mole is hidden underground in a plot of land 18 squares by 18 squares, and you have to find it. You may move North, South, East or West (by pressing N, S, E or W). The mole will send you messages in morse code, to tell you which way to go. See if you can find it in ten moves or less.

The morse code symbols for the four compass points are

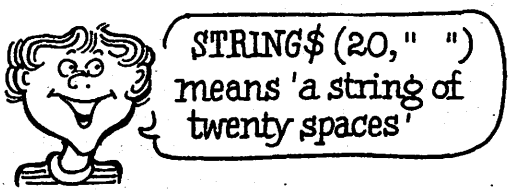
● ● ● ● – ● and ● – –

The mole will give you a short sound for a dot and a longer one for a dash. If you need to go N.E., for example, you will hear the morse for N, followed by the morse for E. I won't tell you which symbol is which compass point. You can work that out from the program!




```
   10 REM DOTTY DIGGER
   20 REM CHARACTERS
   30 VDU 23,240,56,56,16,254,16,40,68,
130
   40 LET YOU$ = CHR$(240)
   50 VDU 23,241,1,6,30,54,60,120,252,
254
   60 LET MOLE$ = CHR$(241)
```

```
   70 REM SCREEN
   80 MODE 5
   90 VDU 23;8202;0;0;0;
  100 COLOUR 129
  110 FOR ROW = 1 TO 18
  120 PRINT TAB(0,ROW) STRING$(20," ")
  130 NEXT ROW
  140 COLOUR 128
  150 PRINT TAB(17,27) "N"; TAB(16,28)
"W#E"; TAB(17,29) "S"
```




```
  160 REM MOLE'S POSITION
  170 LET MC = RND(18)
  180 LET MR = RND(18)

  190 REM YOUR START POSITION
  200 LET YC = RND(18)
  210 LET YR = 18
  220 COLOUR 2: PRINT TAB(YC,YR) YOU$

  230 REM PLAY
  240 FOR M = 1 TO 10
  250 COLOUR 3: PRINT TAB(0,21)
"DIRECTION?"; TAB(0,22) "N,S,E OR W"
  260 INPUT D$
  270 PRINT TAB(0,25) "HOW MANY PACES?"
  280 INPUT P
  290 REM YOUR NEW POSITION
  300 PRINT TAB(YC,YR) " "
  310 IF D$ = "N" THEN LET YR = YR-P
  320 IF YR < 1 THEN LET YR = 1
```

```
  330 IF D$ = "S" THEN LET YR = YR+P
  340 IF YR > 18 THEN LET YR = 18
  350 IF D$ = "W" THEN LET YC = YC-P
  360 IF YC <1 THEN LET YC = 1
  370 IF D$ = "E" THEN LET YC = YC+P
  380 IF YC >18 THEN LET YC = 18
  390 COLOUR 2: PRINT TAB(YC,YR) YOU$
  400 FOR ROW = 21 TO 26
  410 PRINT TAB(0,ROW) STRING$(15," ")
  420 NEXT ROW

  430 REM FOUND THE MOLE?
  440 IF YC = MC AND YR = MR THEN GOTO
610

  450 REM MOLE'S RESPONSE
  460 IF MR < YR THEN SOUND 0,-10,1,9:
SOUND 0,0,1,3: SOUND 0,-10,1,3: SOUND
0,0,1,9
  470 IF MR > YR THEN FOR A = 1 TO 3:
SOUND 0,-10,1,3: SOUND 0,0,1,3: NEXT
A:SOUND 0,0,1,9
  480 IF MC > YC THEN SOUND 0,-10,1,3:
SOUND 0,0,1,9
  490 IF MC < YC THEN SOUND 0,-10,1,3:
SOUND 0,0,1,3: SOUND 0,-10,1,9:SOUND
0,0,1,3: SOUND 0,-10,1,9: SOUND 0,0,1,9
  500 FOR W = 1 TO 3000: NEXT W

  510 NEXT M

  520 REM NO MORE MOVES
  530 FOR W = 1 TO 3000:NEXT W
  540 PRINT TAB(0,25) "TOO LATE!"
  550 PRINT TAB(MC,MR) MOLE$
  560 FOR P = 21 TO 5 STEP -8
  570 SOUND 1,-10,9,20
```

```
580 NEXT P
590 PRINT TAB(0,30)
600 END

610 REM VICTORY
620 PRINT TAB(0,25) "YOU'VE FOUND ME!"
630 FOR W = 1 TO 1000:NEXT W
640 PRINT TAB(MC,MR) MOLE$
650 FOR P = 53 TO 101 STEP 4
660 SOUND 1,-10,P,2
670 NEXT P
680 PRINT TAB(0,30)
```

Variables
String variables

YOU$ is a human figure.

MOLE$ is the mole.

DIRECTION$ is the direction you wish to move in.

Number variables

MC is the mole's column.
MR is the mole's row.
YC is your column.
YR is your row.
P is the number of paces you wish to move.

Changes you can make

If you find the mole hunt too easy (or too difficult!), change the number of moves you can have. This is the M in line 240.

You always start at the south end of the plot of ground. Change line 210 so that the computer chooses at random the row you are to start in.

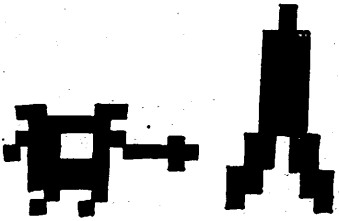
As the program stands, you might start in the same square as the mole, and you wouldn't know it. Do you think the victory sound should play if this happens? Or should the computer go back and choose another position for you? Put in an extra line (215) to solve this problem.

ROCKET LAUNCH

This is a longer program, and uses a number of BASIC keywords that have not been covered in this book. I have included it to show you some of the things you can do when you have learnt more about programming your BBC Micro.

It is a word guessing game, rather like 'hangman'. The computer chooses a word and prints a row of stars to show you how many letters it has. You have to discover what the word is. You may guess one letter at a time, or have a go at guessing the whole word. If you have great difficulty, the computer will give you a clue.

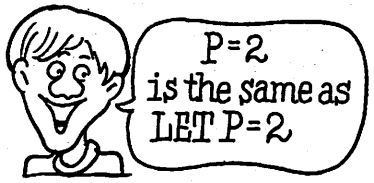
Each time you guess wrong, an alien will move nearer to your space rocket. If you discover the word before he reaches you, the rocket will take off. If not, it will be vapourised!



At the end of the program, from line 930, you will find the ten words the computer chooses from, with their clues. If you want to change the words, just alter these lines. Putting in, on each line, first the word, then a comma, and then the clue.

```
   10 REM ROCKET LAUNCH
   20 REM GRAPHICS
   30 VDU 23,250,16,16,56,56,56,56,56,56
   40 VDU 23,251,56,56,56,108,108,198,
198,130
   50 FOR A = 1 TO 7
   60 READ C
   70 R$ = R$ + CHR$(C)
   80 NEXT A
   90 DATA 250,8,10,251,8,10,32
  100 VDU 23,252,49,31,57,89,31,31,9,18
  110 VDU 23,253,128,0,130,127,2,0,0,0
  120 ALIEN$ = " " + CHR$(252) +
CHR$(253)
  130 VDU 23,254,255,255,0,0,0,0,0,0
  140 MODE 5
  150 VDU 23;8202;0;0;0;

  160 REM START GAME
  170 SCORE = 0: DIM W(10)
  180 FOR A = 1 TO 10: CLS
```




```
  190 P = 2
  200 PRINT TAB(0,0) "Rocket launch"
  210 PRINTTAB(0,29)STRING$(20,
CHR$(254))
  220 PRINT TAB(15,27) R$
  230 PROCALIEN
  240 PROCCHOOSEWORD
  250 REPEAT
  260 PRINT TAB(0,4) "Word ";A
  270 PRINT TAB(0,6) L$
  280 IF P = 10 THEN PROCCLUE
  290 PRINT TAB(0,14) "Enter your
guess."
  300 PRINT TAB(0,16) "The whole word,"
  310 PRINT TAB(0,18) "or just one
letter."
  320 INPUT TAB(0,20) G$
  330 IF G$ ="" THEN 320
  340 FOR B = 14 TO 20 STEP 2
  350 PRINT TAB(0,B) STRING$(20," ")
  360 NEXT B
  370 IF LEN(G$) = 1 THEN
PROCCHECKLETTER ELSE PROCCHECKWORD
  380 PRINT TAB(0,6) L$
  390 UNTIL L$ = W$ OR P = 12

  400 IF L$ = W$ THEN PROCTAKEOFF ELSE
PROCDESTROY
  410 INPUT TAB(0,14) "Press RETURN " I$

  420 NEXT A
```

```
  430 CLS
  440 PRINT TAB(0,10) "Score: ";SCORE;
" out of 10"
  450 PRINT TAB(0,13) "Another game?"
  460 PRINT TAB(0,15) "Y or N"
  470 REPEAT
  480 IF GET$ = "Y" THEN RUN
  490 UNTIL GET$ = "N"
  500 PRINT TAB(0,18) "GOOD-BYE!"
  510 END

  520 DEF PROCCHOOSEWORD
  530 R = RND(10)
  540 IF W(R) = 1 THEN 530
  550 W(R) = 1
  560 RES = 920 + R * 10
  570 RESTORE RES
  580 READ W$,M$
  590 L$ = STRING$(LEN(W$),"*")
  600 ENDPROC

  610 DEF PROCCHECKLETTER
  620 K = 0
  630 FOR B = 1 TO LEN(W$)
  640 IF G$ = MID$(W$,B,1) THEN L$ =
LEFT$(L$,B-1) + G$ +
RIGHT$(L$,LEN(L$)-B): K = K +1: SOUND
1,-10,101,1
  650 NEXT B
  660 IF K = 0 THEN P = P + 1: PROCALIEN
  670 ENDPROC

  680 DEF PROCALIEN
  690 PRINT TAB(P,28) ALIEN$
  700 IF P > 2 THEN SOUND 1,-10,53,1
  710 ENDPROC
```

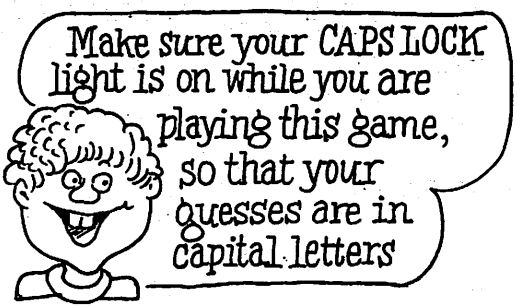
```
720 DEF PROCCHECKWORD
730 IF G$ = W$ THEN L$ = W$: ENDPROC
740 P = P + 1: PROCALIEN
750 ENDPROC

760 DEF PROCTAKEOFF
770 FOR ROW = 27 TO 0 STEP -1
780 PRINT TAB(15,ROW) R$
790 SOUND 1,-10,100-ROW,1
800 FOR W = 1 TO 100: NEXT W
810 NEXT ROW
820 SCORE = SCORE +1
830 ENDPROC

840 DEF PROCDESTROY
850 SOUND 0,-15,3,20
860 PRINT TAB(15,28)"*"
870 PRINT TAB(15,27)"*"
880 PRINT TAB(0,6) W$
890 ENDPROC

900 DEF PROCCLUE
910 PRINT TAB(0,10) "Clue: " M$
920 ENDPROC

930 DATA OPPORTUNITY,CHANCE
940 DATA ARTIFICIAL,FALSE
950 DATA EXTERIOR,OUTSIDE
960 DATA DELICIOUS,TASTY
970 DATA CONCLUSION,END
980 DATA TEMPERATURE,HEAT
990 DATA GENUINE,REAL
1000 DATA IMMENSE,HUGE
1010 DATA OBSTINATE,STUBBORN
1020 DATA DISEASE,ILLNESS
```


Make sure your CAPS LOCK light is on while you are playing this game, so that your guesses are in capital letters

CHAPTER 34

You could do it like this

There are several ways of writing the programs in the 'For you to do' sections of this book. The following are some suggestions.

Chapter 3

```
PRINT "ROGER JONES"
```

Chapter 5

```
PRINT (8 - 2) * 3
```

gives the answer 18.

Chapter 6

PRINT by itself tells the computer to leave one blank line.

Chapter 7

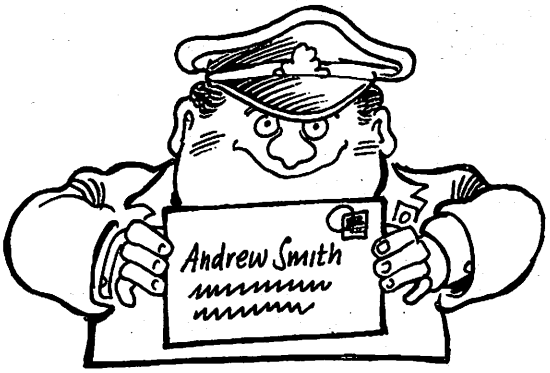
Name

```
10 PRINT "ANDREW SMITH,"
20 PRINT "73 NEW ROAD,"
30 PRINT "ROMTOWN."
40 PRINT "BBC 32K"
```

Chapter 8

Name in lower case

```
10 PRINT "Andrew Smith,"
20 PRINT "73 New Road,"
30 PRINT "Romtown."
40 PRINT "BBC 32K"
```



Chapter 9

MODE 3 has 80 letters on one line. MODE 5 has 20, and MODE 7 has 40.

Chapter 10

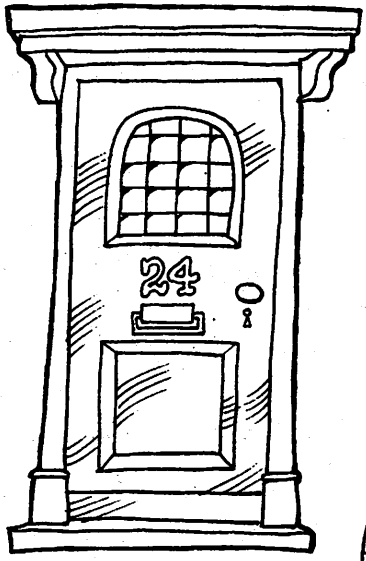
The two lines are:

```
 5 PRINT
25 PRINT
```

Chapter 11

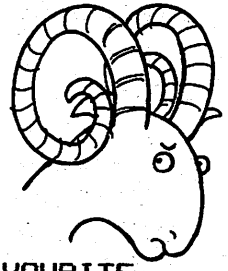
House number

```
10 MODE 7
20 PRINT "WHAT IS THE NUMBER OF YOUR
HOUSE?"
30 INPUT HOUSE
40 PRINT "YOU LIVE AT HOUSE NUMBER "
HOUSE
```



Chapter 14

Animal



```
10 MODE 6
20 PRINT "WHICH IS YOUR FAVOURITE
ANIMAL?"
30 INPUT ANIMAL$
40 PRINT "I DO NOT LIKE THE " ANIMAL$
50 PRINT "I PREFER RAMS"
```

Chapter 15

Birth date

```
10 LET YEAR = 1984
20 CLS
30 PRINT "WHAT IS YOUR AGE"
40 PRINT "ON YOUR BIRTHDAY THIS
YEAR?"
50 INPUT AGE
60 LET BIRTH = YEAR - AGE
70 PRINT "YOU WERE BORN IN " BIRTH
```

Chapter 16

Addition

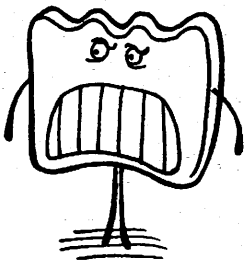
```
10 MODE 7
20 PRINT "GIVE ME A NUMBER"
30 INPUT N1
40 PRINT "GIVE ME ANOTHER NUMBER"
50 INPUT N2
60 LET A = N1 + N2
70 PRINT "YOUR TWO NUMBERS ADDED
TOGETHER MAKE" A
```

Chapter 18

Clever computers

```
10 CLS
20 PRINT "WHAT IS YOUR NAME?"
30 INPUT NAME$
40 PRINT "DO YOU THINK COMPUTERS ARE
CLEVER?"
50 INPUT ANSWER$
```

```
   60 IF ANSWER$ = "YES" THEN PRINT
NAME$ " YOU ARE FANTASTIC" ELSE PRINT
"I'M DISGUSTED WITH YOU, " NAME$
```



Chapter 19

The two lines to be changed are:

```
   20 LET NUMBER = RND(12)
   30 PRINT "MY NUMBER IS BETWEEN 1 AND
12"
   70 PRINT "MY NUMBER WAS ";NUMBER
```

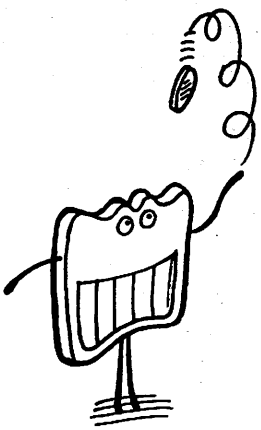
Chapter 20

Multiplication

```
10 CLS
20 LET A = RND(100)
30 LET B = RND(100)
40 LET C = A * B
50 REPEAT
60 PRINT "WHAT IS "; A ;" TIMES "; B
70 INPUT D
80 UNTIL D = C
90 PRINT "RIGHT.  WELL DONE."
```

Chapter 21

Toss a coin



```
  10 CLS
  15 REPEAT
  20 LET N = RND(2)
  30 IF N = 1 THEN LET S$ = "HEADS"
  40 IF N = 2 THEN LET S$ = "TAILS"
  50 PRINT "HEADS or TAILS?"
  60 INPUT A$
  70 IF A$ = S$ THEN PRINT "RIGHT" ELSE
PRINT "WRONG"
  80 UNTIL A$ = "STOP"
```

Chapter 22

16 times table

```
  10 CLS
  20 FOR N = 1 TO 20
  30 PRINT N; "*16="; N*16
  40 FOR C = 1 TO 3000
  50 NEXT C
  60 NEXT N
```

Chapter 23

ASCII codes

```
10 MODE 4
20 PRINT "ASCII CODE  CHARACTER"
30 FOR N = 123 TO 126
40 PRINT N;  " ";CHR$(N)
50 NEXT N
```

RUN the program. Then change line 10 to

```
10 MODE 7
```

and RUN the program again.

Chapter 24

Planes

```
   10 VDU 23,225,16,24,156,255,156,24,16
,0
   20 LET PLANE$ = CHR$(225)
   30 MODE 6
   40 FOR C = 1 TO 320
   50 PRINT " " PLANE$ " ";
   60 NEXT C
```

Chapter 25

Alien

```
   10 VDU 23,224,124,84,124,124,40,40,68
,0
   20 MODE 5
   30 PRINT TAB(9,16) CHR$(224)
```

Chapter 26

Rocket

```
10 MODE 4
20 VDU 23,235,7,31,126,252,126,31,7,0
30 LET ROCKET$ = CHR$(235) + " "
40 FOR COLUMN = 38 TO 0 STEP - 1
50 PRINT TAB(COLUMN,16) ROCKET$
60 FOR W = 1 TO 100
70 NEXT W
80 NEXT COLUMN
```

Chapter 27

Moving name

```
10 MODE 5
20 COLOUR 131: CLS
30 COLOUR 1
40 FOR COLUMN = 1 TO 12
50 PRINT TAB(COLUMN,10) " CAROLYN"
60 FOR W = 1 TO 1000
70 NEXT W
80 NEXT COLUMN
```

Chapter 28

Saucer

```
   10 MODE 4
   20 LET FS$ = "_/*\_"
   30 PRINT TAB(0,25);"_____________________
_____________________________"
   40 FOR ROW = 0 TO 24
   50 PRINT TAB(18,ROW);FS$
   55 SOUND 0,-10,1,5
   60 FOR W = 1 TO 500
   70 NEXT W
   80 IF ROW < 24 THEN PRINT
TAB(18,ROW);"      "
   90 NEXT ROW
```

Line 55 makes the sound.

Chapter 29

Copter

```
   10 MODE 5: COLOUR 131: CLS
   20 VDU 23;8202;0;0;0;
   30 VDU 23,250,130,124,146,16,124,255,
126,0
   40 LET COPTER$ = CHR$(250)
   50 LET COPCOL = RND(18)
   60 LET PADCOL = RND(18)
   70 COLOUR 0:PRINT TAB(0,28);
"~~~~~~~~~~~~~~~~~~~~~~~~~~~~"
```

```
   80 COLOUR 1: PRINT TAB(PADCOL,28);"-"
   90 FOR ROW = 0 TO 28
  100 COLOUR 2: PRINT TAB(COPCOL,ROW);
COPTER$
  105 SOUND 0,-10,1,1
  110 LET DIRECTION$ = INKEY$(10)
  120 IF ROW < 28 THEN PRINT
TAB(COPCOL,ROW);" "
  130 IF DIRECTION$ = "X" AND COPCOL <
19 THEN LET COPCOL = COPCOL + 1
  140 IF DIRECTION$ = "Z" AND COPCOL > 0
THEN LET COPCOL = COPCOL - 1
  150 NEXT ROW
  160 COLOUR 0: PRINT TAB(0,30) "PLAY
AGAIN? (Y OR N)"
  170 REPEAT
  180 LET A$ = GET$
  190 IF A$ = "Y" THEN RUN
  200 UNTIL A$ = "N"
  210 PRINT "GOODBYE"
```

Lines 10, 70, 80 and 100 have been changed to put in the COLOUR commands. Line 105 makes the sound as the helicopter comes down. Lines 160 to 200 ask you if you would like to play again.

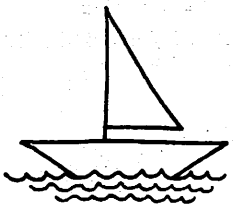
Chapter 30

Here is the face:

```
 10 MODE 5: GCOL0,130: CLG: GCOL0,1
 20 MOVE 500,200
 30 DRAW 500,300
 40 DRAW 300,300
 50 DRAW 300,700
 60 DRAW 900,700
 70 DRAW 900,300
 80 DRAW 700,300
 90 DRAW 700,200
100 MOVE 400,500
110 DRAW 400,600
120 DRAW 500,600
130 DRAW 500,500
140 DRAW 400,500
150 MOVE 700,500
160 DRAW 700,600
170 DRAW 800,600
180 DRAW 800,500
190 DRAW 700,500
200 MOVE 500,400
210 DRAW 700,400
```

This is the yacht:

```
 10 MODE 5: GCOL0,131: CLG
 20 GCOL0,0
 30 MOVE 300,300
 40 DRAW 600,300
 50 PLOT 85,400,200
 60 PLOT 85,800,200
 70 DRAW 900,300
 80 PLOT 85,600,300
 90 MOVE 500,300
100 DRAW 500,800
110 GCOL0,1
120 DRAW 800,400
130 PLOT 85,500,400
```



Chapter 31

Three mice

Here are the changes to the 'right' and 'wrong' parts of the program:

```
310 REM WRONG
315 FOR COLUMN = 0 TO 17
320 PRINT TAB(COLUMN,16);" " MOUSE$
322 FOR W = 1 TO 50: NEXT W
325 NEXT COLUMN
330 SOUND 1,-15,5,40
340 END
350 REM RIGHT
360 FOR A = 1 TO 300
370 COLOUR RND(3)
380 PRINT MOUSE$;
390 NEXT A
400 SOUND 1,-15,53,10
410 SOUND 1,-15,101,10
```

Index

Adaptor – electrical 12
Adding words 51
Addition 20
Aerial lead 13
AND 117
Arrow keys 56
ASCII code 79
ASC 79

Background colours 101
BASIC keywords 11
Blank lines of print 28
Boxes for numbers 38
Boxes for strings 46
Brackets 22
BREAK 29

Caps lock 13
Cassette leads 134
Cassette motor 15
Cassette recorder 133
Catalogue of programs 136
CHAIN 138
Changing a program 56
Channels for sound 106
Characters 45
Chords 108
CHR$ 79
Clearing the graphics screen 125
Clearing the memory 29
Clearing the screen 35
CLG 125
CLS 35
Code numbers 79
COLOUR 101
Coloured lines 124
Columns 90
Commas 43
COPY 56
Co-ordinates 91
Counting backwards 99
CTRL N 132
CTRL O 132
Cursor 14

Delete key 18
Designing a character 82
Division 21
Documentation 128
Dollar sign 46
DRAW 120
Drawing a square 121
Drawing lines 121

Editing mode 57
ELSE 61
Empty string 45
Equals sign 49
Error messages 19
ESCAPE 64

Fields 44
File names 135
Filling a triangle 122
Flashing colours 104
Flying saucer program 98
FOR . . . NEXT loops 75
Foreground colours 102

GCOL0 124
GET$ 112
GOTO 63
Graphics characters 82
Graphics cursor 120
Guess the number program 65

Helicopter program 114
Hi-res graphics 119

IF 60
INKEY$ 112
INPUT 36
Invader character 83
Is less than sign 117
Is more than sign 74
Is not equal to sign 70

Keyboard 15

Leaving a program 64
LET 49
Line numbers 24
LIST 26
LOAD 137
Loops 66
Lower case letters 31

Mathematical calculations 20
Memory 11
Mistakes 18
MODEs 32
Motorcycle noise 110
Mouse character 129
MOVE 120
Moving characters 93
Moving the T.V. display 33
Multiplication 21
Multiplication tables 76
Musical notes 106

Nested loops 77
NEW 29
Noises 109
Null string 45
Number variables 39

OLD 30

Paged mode 132
Pauses 76
Pitch 106
Pixel 83
Plane character 87
PLOT 85 122
PRINT 16
Printing numbers 41
Priorities 22
Programs 24
Prompt 14

Quick reactions 112
Quotation marks 16

RAM 11
Random numbers 63
Random words 70
Read cursor 57
Recording a program 133
Recovering a program 30
Red lights 15
REM 128
Remarks 128

Reminder 128
Remote socket 134
REPEAT 66
REPEAT . . . UNTIL loops 66
RETURN 16
RND 61
Rocket character 99
ROM 11
Rows 89
RUN 25

SAVE 135
Scroll 132
Semi-colons 42
Shift 16
Shift lock 15
SOUND 106
Sound effects 106
Spaces 27
STEP 99
Storing numbers 38
Storing words 46
String variables 46
Strings 45
Subtraction 21

T.V. connections 13
TAB 91
Teletext 32
THEN 60
Three mice program 126
TO 75
Truck program 93
Turn off cursor 115

UHF socket 13
UNTIL 66
Upper case letters 31
Uses of computers 9

Variable names 53
VDU 23 85

Write cursor 57
*CAT 136
*TV 33
= 49
<> 70
< 74
> 117
" 16
$ 46

MODE details

Screen size

MODE	*Columns*	*Rows*
0	80	32
1	40	32
2	20	32
3	80	25
4	40	32
5	20	32
6	40	25
7	40	25

Colour and Graphics

MODE	*No. of colours*	*'Hi-Res' Graphics*
0	2	yes
1	4	yes
2	16	yes
3	2	no
4	2	yes
5	4	yes
6	2	no
7	*	no

*MODE 7 can show different colours, but not by using the word COLOUR.

Colour Numbers

MODEs *0, 3, 4 and 6*

Colour	*Foreground*	*Background*
black	0	128
white	1	129

MODEs *1 and 5*

Colour	*Foreground*	*Background*
black	0	128
red	1	129
yellow	2	130
white	3	131

MODE *2*

Colour	*Foreground*	*Background*
black	0	128
red	1	129
green	2	130
yellow	3	131
blue	4	132
magenta	5	133
cyan	6	134
white	7	135
flashing black-white	8	136
flashing red-cyan	9	137
flashing green-magenta	10	138
flashing yellow-blue	11	139
flashing blue-yellow	12	140
flashing magenta-green	13	141
flashing cyan-red	14	142
flashing white-black	15	143

Fontana Computer Books

Fontana Paperbacks has a growing list of exciting computer books

New and forthcoming titles include:

First Steps with your Spectrum *Carolyn Hughes*
Second Steps with your Spectrum *Carolyn Hughes*
Elementary Pascal *Henry Legard and Andrew Singer*
Elementary Basic *Henry Legard and Andrew Singer*
Better Programming for Your Spectrum and ZX81
S. Robert Speel
New Adventure Systems for the Spectrum *S. Robert Speel*
The Good Software Guide (Sinclair machines)
The Good Software Guide (VIC 20 and Commodore 64)
Choosing and Using a Microcomputer
Alan Radnor and Howard Kahn
Solving Business Problems
by Computer
Kate Behan and Diana Holmes
Computer Programming for Beginners *Robert Erskine*
Fontana's Dictionary of Computer Terms
Better Programming for Your Commodore 64
Henry Mullish and Dov Kruger
Tim Hartnell's Giant Book of Spectrum Games
Tim Hartnell's Giant Book of Computer Games
Giant Book of Games for Your BBC Micro *ed. Tim Hartnell*
Giant Book of Games for Your Dragon *ed. Tim Hartnell*
Giant Book of Games for Your VIC 20 *ed. Tim Hartnell*
Giant Book of Games for Your Commodore 64
ed. Tim Hartnell

First Steps with your Spectrum

A Beginner's Guide to Programming

by Carolyn Hughes

This book takes you step by step from your first tap at the keyboard to writing your own Spectrum programs.

DISCOVER how to use colour and sound
MAKE objects move about the screen
LEARN BASIC programming techniques
IMPROVE your skills with specially written programs

PLUS six exciting games programs, including *Morse Mole, Mind Reader* and *Alien.*

". . .eases the earliest beginner into BASIC comfortingly and successfully." *Guardian*
". . .concise and lucid explanations." *T.E.S.*

FONTANA PAPERBACKS

An exciting new idea for young programmers!

The Bytes Brothers

Solve-It-Yourself Computer Mysteries

by Lois & Floyd McCoy

Now young computer fans can become detectives and use a microcomputer to solve thrilling mysteries.

Brent and Barry Byte are brilliant young programmers who, with Nibble, their family micro, investigate a series of baffling cases. Each book contains five mysteries with an integral computer program which the reader (and the Bytes Brothers) types in to discover the solution.

The programs are accompanied by useful "debugging" sections which explain how they work and, where necessary, how they can be adapted for different computers.

ARMADA PAPERBACKS

Fontana Paperbacks: Non-fiction

Fontana is a leading paperback publisher of non-fiction, both popular and academic. Below are some recent titles.

- ☐ WHAT DO WOMEN WANT? Luise Eichenbaum and Susie Orbach £1.75
- ☐ AVALONIAN QUEST Geoffrey Ashe £2.50
- ☐ WAR AND SOCIETY IN EUROPE, 1870–1970 Brian Bond £3.50
- ☐ MARRIAGE Maureen Green £2.75
- ☐ AMERICA AND THE AMERICANS Edmund Fawcett and Tony Thomas £2.95
- ☐ FRANCE, 1815–1914: THE BOURGEOIS CENTURY Roger Magraw £4.95
- ☐ RULES OF THE GAME Nicholas Mosley £2.50
- ☐ THIS IS THE SAS Tony Geraghty £4.95
- ☐ THE IMPENDING GLEAM Glen Baxter £2.95
- ☐ A BOOK OF AIR JOURNEYS Ludovic Kennedy (ed.) £3.95
- ☐ TRUE LOVE Posy Simmonds £2.95
- ☐ THE SUPPER BOOK Elizabeth Kent £2.95
- ☐ THE FONTANA BIOGRAPHICAL COMPANION TO MODERN THOUGHT Alan Bullock and R. B. Woodings (eds.) £6.95
- ☐ LIVING WITH LOSS Liz McNeill Taylor £1.75
- ☐ RASPUTIN Alex de Jonge £2.95
- ☐ THE PRIVATE EYE STORY Patrick Marnham £4.95
- ☐ BARTHES: SELECTED WRITINGS Susan Sontag (ed.) £4.95
- ☐ A POCKET POPPER David Miller (ed.) £4.95
- ☐ THE WRITINGS OF GANDHI Ronald Duncan (ed.) £2.50

You can buy Fontana paperbacks at your local bookshop or newsagent. Or you can order them from Fontana Paperbacks, Cash Sales Department, Box 29, Douglas, Isle of Man. Please send a cheque, postal or money order (not currency) worth the purchase price plus 15p per book for postage (maximum postage required is £3).

NAME (Block letters) _____

ADDRESS _____

_____

While every effort is made to keep prices low, it is sometimes necessary to increase prices at short notice. Fontana Paperbacks reserve the right to show new retail prices on covers which may differ from those previously advertised in the text or elsewhere.

GET TO KNOW THE AMAZING BBC MICRO...

This book will show you, step by step, how to use one of the fastest, most powerful home computers. Here is all you need to take you from your first tap at the keyboard to writing your own programs.

DRAW pictures and diagrams
MAKE objects move about the screen
EXPERIMENT with colour and sound
LEARN BASIC programming techniques
DEVELOP your skills with specially written programs

PLUS exciting games programs for beginners...
COPTER Guide the helicopter safely over the sea to its landing pad.
THREE MICE Memorize the colour of the mice – before they disappear!
JACKPOT! A game that turns your micro into a fruit machine. Trap the stars and you could be lucky!
DOTTY DIGGER Unearth the missing mole by tracking its morse code messages.
ROCKET LAUNCH Guess the hidden word before the alien vaporizes your space rocket!

ISBN 0-00-692365-8

U.K. £1.50